Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Andrej Jugovic

# Abstraktivno povzemanje dokumentov v slovenskem jeziku

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: izr. prof. dr. Marko Robnik-Šikonja

Ljubljana, 2016

*Besedilo je oblikovano z urejevalnikom besedil LaTeX.*

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Za mnoge tematike obstaja velika množica dokumentov, iz katerih človek, že zaradi ogromne količine besedila, le težko izlušči pomembne informacije. Kot pomoč pri tem opravilu se je razvilo področje avtomatskega povzemanja, ki s pomočjo tehnik obdelave naravnega jezika avtomatsko izdela povzetke. Večina dela je opravljenega za angleški jezik. Izdelajte prototip sistema za avtomatsko abstraktivno povzemanje besedil v slovenščini. Uporabite obstoječa orodja za jezikovno analizo pri tvorbi semantičnih trojic, ki jih povežite v graf. Določite najpomembnejša vozlišča grafa in na tej podlagi generirajte povzetke. Razvito metodo ovrednotite na učnem korpusu povzetkov, ki ga pridobite iz Wikipedije.

*Za vse nasvete in strokovno pomoč se zahvaljujem izr. prof. dr. Marku Robniku-Šikonji.*

*Zahvaljujem se očetu Cirilu, materi Tatjani, bratoma Matjažu in Alešu, ter vsem prijateljem za vso potrpljenje in spodbudo tekom študija*

# Kazalo

# Seznam uporabljenih kratic

| kratica | angleško | slovensko |
|---|---|---|
| **BM** | Best matching | Najboljše ujemaje |
| **DSYNT** | Deep-Syntactic tree | Globoko sintaktično drevo |
| **FD** | Functional descriptions | Funkcijski deskriptorji |
| **FUF** | Functional unification formalism | Funkcijska unifikacija formalizmov |
| **IDF** | Inverse document frequency | Inverzna pogosotost v dokumentih |
| **NLP** | Natural Language Processing | Procesiranje naravnega jezika |
| **NLTK** | Natural Language Toolkit | Knjižnica NLTK za procesiranje naravnega jezika |
| **MSTParser** | Minimum-Spanning Tree Parser | Razčlenjevalnik z minimalnim vpetim drevesom |
| **PR** | PageRank algorithm | Algoritem PageRank za rangiranje vozlišč v grafu |
| **P-PR** | Personalized PageRank algorithm | Algoritem personalized PageRank za rangiranje vozlišč v grafu |
| **ROUGE** | Recall-oriented understudy for gisting evaluation with n-grams | Mera za ocenjevanje kakovosti povzetkov, ki temelji na ujemanju besed |
| **synset** | Synset | Zbirka sopomenk |
| **TF** | Term frequency | Pogostost besede |

# Povzetek

**Naslov:** Abstraktivno povzemanje dokumentov v slovenskem jeziku

V diplomskem delu obravnavamo avtomatsko povzemanje slovenskih dokumentov. Živimo v času, ko imamo na voljo veliko dokumentov v elektronski obliki, ki jih želimo strniti v krajše zapise. Vseh ne moremo ročno povzeti, zato je potrebno postopek avtomatizirati.

S pomočjo razčlenjevalnika za slovenski jezik smo v dokumentu poiskali trojice, sestavljene iz osebka, povedka in predmeta. Iz besed, ki so v teh trojčkih, smo zgradili graf in povezave v grafih utežili na različne načine. Vozlišča v grafu smo ocenili z algoritmom P-PR. To nam je služilo kot osnovna ocena pomembnosti besed v trojčkih. P-PR vrednosti besed v trojčkih smo utežili z merami TF-IDF, Okapi BM-25 in frekvenco besed. S pomočjo teh ocen smo izbrali najboljše trojčke in iz njih generirali povzetke. Dobljene povzetke smo ocenili z merama ROUGE-N in ROUGE-S. Evalvacijo smo izvedli na korpusu, ki smo ga zgradili s pomočjo Wikipedije, in z ročno povzetimi besedili. Rezultati so pokazali, da človek ustvari precej boljše povzetke. Najbolje se je izkazal sistem, kjer smo povezave grafa utežili s številom pojavitve dvogramov, P-PR vrednost pa s frekvenco pojavitve besede v trojicah.

**Ključne besede:** procesiranje naravnega jezika, povzemanje dokumentov, algoritem P-PR (personalizirani PageRank) za rangiranje vozlišč v grafu, mera ROUGE, avtomatsko povzemanje dokumentov.

# Abstract

**Title:** Abstractive summarization for Slovene language

The thesis focuses on automatic summarization of Slovene documents. There are large numbers of documents in digital form which we want to summarize in order to make them accessible to humans. This cannot be done manually so we want to automate the process.

Our system, uses a parser for Slovene language to find triplets consisting of a subject, predicate (or verb) and object. We build a graph using the words in the triplets and weight the connections. We rank the nodes with P-PR algorithm, which assesses the importance of words in triples. We weight P-PR values of words in the triples with measures TF-IDF, Okapi BM-25, and word frequency. We chose the best triplets and use them to generate summaries. Generated summaries are evaluated with ROUGE-N and ROUGE-S measures. Evaluation is performed on a corpus, built from Wikipedia, and also with manually created summaries. The results show that humans create significantly better summaries. The best computer generated summaries are created when graph connections are weighted with the number of bigram occurrences and P-PR values are weighted with the frequency of word occurrence in triplets.

**Keywords:** natural language processing, document summarization, personalized PageRank algorithm, ROUGE measure, weighted links, automatic document summarization.

# Poglavje 1

# Uvod

Živimo v času, ko so na voljo številni dokumenti v elektronski obliki. Število dokumentov se predvsem na spletu iz dneva v dan hitro povečuje. Če bi želeli vse te dokumente prebrati, bi bilo to časovno neizvedljivo. Tako se je pojavila potreba po avtomatskem povzemanju dokumentov. Povzemanje dokumentov je proces, pri katerem želimo osnovni dokument spremeniti v krajšo obliko. Dober povzetek naj bi bil kratek in informativen, Povzetke lahko napravimo iz enega ali več dokumentov, ki opisujejo isti dogodek.

Povzemanje dokumentov spada na področje procesiranja naravnega jezika, ki je podpodročje umetne inteligence in računalniške lingvistike. Ukvarja se z interakcijo med računalnikom in človekom, predvsem z razumevanjem človekovega jezika. Nekateri problemi, s katerimi se področje, poleg avtomatskega povzemanja, še ukvarja, so strojno prevajanje, razčlenjevanje stavkov, razumevanje govora, ustvarjanje govora, iskanje pomena besed, razumevanje besed in odgovarjanje na vprašanja.

Povzemanje dokumentov ločimo na ekstraktivno in abstraktivno. Pri ekstraktivnem povzemanju iz osnovnega dokumenta izberemo nekatere besede, fraze ali stavke, iz katerih nato tvorimo povzetke, pri tem pa izbranih besed, fraz in stavkov ne spremenimo. Največji problem takega povzemanja je, da ne znamo zgraditi novih besed ali stavkov, kot to storijo ljudje. Ta problem poskušamo rešiti pri abstraktivnem povzemanju. Algoritem abstraktivnega

povzemanja najprej zgradi lastno predstavitev dokumenta in iz nje zgradi povzetek s pomočjo metod procesiranja naravnega jezika. Abstraktivno povzemanje je zahtevnejše od ekstraktivnega.

V diplomskem delu se posvetimo abstraktivnemu povzemanju dokumentov, ki ga bomo bolj podrobno opisali v naslednjem poglavju. Na kratko pa je naš postopek sledeč. Iz dokumenta najprej pridobimo semantične trojice, iz katerih zgradimo predstavitev besedila v obliki grafa, v katerem ocenimo pomembnosti vozlišč z algoritmom P-PR (angleško Personalized PageRank) in iz najpomembnejših vozlišč ustvarimo povzetke.

Diplomsko delo je razdeljeno na 6 poglavij. V drugem poglavju opišemo področje abstraktivnega povzemanja, v tretjem so predstavljene pomembnejše tehnologije, ki so uporabljene v našem sistemu. Četrto poglavje govori o tem, kako je sestavljen naš sistem, v petem predstavimo rezultate evalvacije našega modela, v zadnjem pa strnemo ugotovitve in predstavimo možne izboljšave v prihodnosti.

# Poglavje 2

# Avtomatsko abstraktivno povzemanje

Abstraktivno povzemanje na začetku zgradi predstavitev dokumenta in iz nje s pomočjo metod procesiranja naravnega jezika zgradi povzetek. Tako povzemanje je bolj zahtevno kot ekstraktivno povzemanje, saj se v povzetku lahko pojavijo tudi nove besede in besedne fraze, ki jih ni v izvornem dokumentu.

V splošnem abstraktivno povzemanje delimo v dve skupini:

1. strukturen način povzemanja in
2. semantičen način povzemanja.

Strukturen način povzemanja pridobi najpomembnejše informacije s pomočjo kognitivnih shem [14], kot so predloge ali ekstrakcijska pravila. Te metode ustvarijo za angleški jezik dokaj kvalitetne povzetke [18], vendar je lingvistična kvaliteta povzetkov nizka, saj vsebujejo številne gramatične napake.

Pri semantičnem načinu poskušamo dokument semantično predstaviti in nato uporabiti v generatorju za naravni jezik. Metoda je osredotočena na identificiranje samostalniških in glagolskih fraz, ki jih pridobimo s procesiranjem lingvističnih podatkov [38, 18]. Te metode so odvisne od semantične predstavitve izvornega dokumenta [18]. Ustvarijo dokaj dobre, strnjene in informativne povzetke, ki imajo manj lingvističnih napak, kot povzetki ustvar-

jeni s strukturnim načinom povzemanja, vendar so pristopi precej bolj zapleteni.

## 2.1 Strukturni način povzemanja dokumentov

Najbolj znane metode strukturnega načina povzemanja so predloge, ekstrakcijska pravila in druge strukture, kot na primer drevesa in ontologije [18]. Nekatere bomo predstavili v tem razdelku.

Drevesne metode predstavijo vsebino našega dokumenta kot odvisna drevesa. Iz predstavitve ustvarimo povzetke z metodami za generiranje besedila. V primerjavi z ekstraktivnim povzemanjem se izboljša berljivost besedila in zmanjša število redundantnih informacij. Problem teh metod je, da ne upoštevajo konteksta stavka [18]. Primer take metode je metoda zlivanja informacij iz različnih dokumentov. Avtorji članka z naslovom „Information fusion in the context of multi-document summarization“ s to metodo povzemajo časopisne članke, ki opisujejo isti dogodek [2]. S to metodo na začetku poiščemo stavke, ki so si tematsko podobni. Iz njih nato zgradimo globoko sintaktično drevo (DSYNT) [20], ki opisuje relacije med besedami ter vsako besedo semantično opiše. Vozlišča v drevesu hranijo gramatične lastnosti posamezne besede. Med seboj so povezana z naslednjo besedo v stavku. Za tem rekurzivno primerjamo glagole v drevesih. Če imata dve vozlišči enak glagol, ta glagol dodamo v končno drevo ter nadaljujemo s primerjanjem naslednikov obeh dreves. V primeru, da so vsi nasledniki enaki, gre fraza, sestavljena iz drevesa, v presek teme. Sicer primerjamo drevesi s parafraziranimi pravili, ki so vnaprej določena glede na našo podatkovno zbirko. S parafraziranimi pravili poskušamo zajeti možnosti, kako človek na različne načine pove isto frazo. Pravila upoštevajo, da so lahko besede v frazi različno razporejene, imajo različne gramatične lastnosti ali da so opisane s sopomenko. Če s pomočjo parafraziranih pravil določimo, da sta frazi enaki, končno drevo dodamo v presek teme. Po končani primerjavi

stavkov, uredimo preseke po pogostosti pojavitve. V tematski presek dodamo vse preseke nad določenim pragom pojavitve. Izbrane preseke uredimo po časovnem dogajanju s pomočjo časovnih zaimkov. Iz izbranh presekov generiramo besedilo s FUF/SURGE [8, 36] algoritmom. FUF uporablja unifikacijo grafa pri kombiniranju vhodnih struktur v končne stavke. Vhodne strukture so sestavljene iz sintaktično označenih stavkov za določen jezik. Strukture so predstavljene kot funkcijski deskriptorji (FD). Pri besedilih v angleščini za sintaktično označevanje funkcijskih deskriptorjev uporabimo SURGE. SURGE je sintaktični realizator za slovnico. Sintaktično označene funkcijske deskriptorje lineariziramo, da iz njih pridobimo željene stavke.

Dokumenti na spletu so velikokrat domensko povezani, saj govorijo o isti temi ali pa o istem dogodku. To izkoristimo pri metodah, ki so osnovane na ontologijah, tako da pri generiranju povzetkov uporabimo dodatno znanje o nekem področju. Ontologija je s pomočjo entitet strukturirana predstavitev teksta. Entitete so med seboj povezane z relacijami, ki veljajo med njimi npr. entiteta „Stanko Bloudek“ je v relaciji „se je rodil“ z entiteto „Idrija“. Ta način generiranja izboljša kvaliteto povzetkov za specifično domeno, vendar moramo ročno zgraditi slovar entitet in definirati domeno, kar je lahko časovno zelo potratno in neučinkovito. V delu „A fuzzy ontology and its application to news summarization“ [23] so avtorji z modelom mehkih ontologij iz časopisnih člankov v kitajskem jeziku ustvarili povzetke. Avtorji so skupaj z domenskimi strokovnjaki zgradili domensko ontologijo z vnaprej določenimi dogodki in slovar kitajskih novic. S pomočjo kitajskega slovarja novic in preiskovalnim agentom so predprocesirali novice, iz katerih so zgradili mehke ontologije, ki so zgrajene iz mehkih konceptov. Vsak mehki koncept ima množico povezav z različnimi pomembnimi dogodki, definiranimi v domenski ontologiji. Z zgrajeno mehko ontologijo so lahko za novo predprocesirano novico zgradili besedne zveze. S pomočjo algoritma za povzemanja stavčnih poti iz besednih zvez, so pridobili vse mogoče stavčne poti, določene v mehkih ontologijah. S stavčnim generatorjem so nato ustvarili povzetke iz pridobljenih stavčnih poti. Stavke so še prečistili, nekatere izbrisali ter ustva-

rili povzetek.

Ontologije so prilagojene določeni domeni. Za splošne domene se je poizkušalo zgraditi modele povzemanja, ki delujejo z vnaprej določenimi pravili, na katere mora model odgovoriti. Za te metode je značilno, da so dokumenti predstavljeni kot izrazi kategorij in kot seznami vidikov. Stavki so ustvarjeni v skladu s povzemalnimi pravili, ki odgovorijo na različne vidike, določene v kategorijah in se generirajo s pomočjo vzorcev [18]. Te metode imajo podoben problem kot metode, ki temeljijo na ontologijah: potrebno je vnaprej ročno zgraditi pravila, kar je časovno potratno in zahtevno delo, vendar dobimo za angleški jezik precej kratke in berljive povzetke. Primer take metode je algoritem, opisan v članku „Fully abstractive approach to guided summarization“ [15], kjer so avtorji poskušali napraviti kratke povzetke z vodenim povzemanjem. Delo sloni na povzemanju posameznih kategorij, z vnaprej pripravljenim seznamom vidikov, za katere mislimo, da moramo nanje odgovoriti v končnem povzetku. Začeli so s predprocesiranjem vseh besedil. Med predprocesiranjem so besedilo normalizirali z lematizacijo besed ter segmentacijo in tokenizacijo stavkov. Iz tega so ustvarili abstraktne sheme, ki temeljijo na ekstrakciji informacij, hevristiki za izbiro vsebine ter generatorskih vzorcih. Vsaka shema je ustvarjena, da odgovori na eno ali več vprašanj nekega vidika v kategoriji. Nekateri vidiki imajo lahko več shem. Pravila za povzemanja informacij ponavadi vrnejo več kandidatov za vsak vidik. Med njimi izberejo tistega, ki je za nek vidik največkrat omenjen. Na koncu so izbrane kandidate, s pomočjo vnaprej pripravljenih povzemalnih vzorcev ustvarjenih za neko temo, uredili v povzetek.

## 2.2 Semantično povzemanje dokumentov

Metode semantičnega povzemanja dokumentov uporabljajo pri ustvarjanju povzetkov semantično predstavljen dokument. Najbolj znane metode so multimodalni semantični model, metode bazirane na delu informacije (angl. information items) ter metode s semantičnimi grafi.

Z metodami, ki temeljijo na delu informacije, gradimo povzetke iz abstraktne predstavitve dokumenta namesto iz osnovnih stavkov dokumenta. Abstraktna predstavitev je sestavljena iz najmanjših elementov povezanih informacij v besedilu [18]. Najmanjše povezane informacije v besedilu so lahko vse od lastnosti entitete pa do opisa celotnega dogodka ali akcije. Metode ustvarijo kratke povzetke, ki so lahko lingvistično precej slabe kvalitete, če izberemo nepravilen razčlenjevalnik osnovnega dokumenta [18]. Primer uporabe take metode opisuje članek „Framework for abstractive summarization using text-to-text generation“ [14]. Deli informacij so predstavljeni kot datumsko in prostorsko določene trojice osebek-povedek-predmet, ki jih dobimo s sintaktično analizo in lingvističnimi anotacijami. Iz teh trojic so generirani stavki. Med vsemi generiranimi stavki so izbrani tisti, ki imajo največje povprečje dokumentne pojavitve, ki je izračunano tako, da je za vsako besedo v ustvarjenem stavku sešteto število pojavitev v besedilih, iz katerih povzemamo, ter deljeno s številom besed v stavku. Povzetek ustvarimo iz izbranih stavkov. Pri temu so upoštevani datumi in lokacije, kjer se je dogodek, opisan v stavku, zgodil.

Med metode semantičnega načina povzemanja dokumentov spadajo tudi metode, ki gradijo povzetke s pomočjo semantičnih grafov. Te metode zgradijo semantične grafe iz originalnega dokumenta. Graf zatem zreduciramo ter iz njega ustvarimo povzetek. Pridobljeni povzetki so jedrnati in gramatično pravilni. Primer takega načina povzemanja je opisan v članku „Semantic graph reduction approach for abstractive text summarization“ [31], kjer so avtorji ustvarili bogate semantične grafe. V semantičnih grafih so v vozliščih glagoli in samostalniki osnovnega dokumenta. Povezave med vozlišči predstavljajo semantične in topološke relacije. Pri predstavitvi vozlišč in glagolov so si pomagali z domensko ontologijo. Ker je graf predstavljen z domensko ontologijo, je zmožen hraniti pomen besed, stavkov in odstavkov. Zgrajeni graf je zreduciran s pomočjo hevrističnih pravil. Graf so zreducirali z menjavo, brisanjem ali popravljanjem vozlišč grafa s pomočjo WordNet [30] relacij. Iz zreduciranega semantičnega grafa so nato ustvarili stavke za povzetek.

# Poglavje 3

# Jezikovni viri in tehnologije

V tem poglavju bomo opisali vire in tehnologije, ki smo jih uporabili v našem sistemu za avtomatsko povzemanje. Opisali bomo razčlenjevalnik za slovenski jezik, algoritem P-PR za rangiranje vozlišč v grafu, semantično vektorsko predstavitev besed word2vec, slovenski korpus ccGigafida, bazo besednih relacij WordNet in njeno slovensko različico SloWNet.

## 3.1 Razčlenjevalnik za slovenski jezik

Razčlenjevalnik za slovenski jezik [6] je računalniški program, ki je bil razvit med letoma 2008 in 2013 v okviru projekta Sporazumevanje v slovenskem jeziku (`http://www.slovenscina.eu/`). V sklopu projekta se je poleg razčlenjevalnika razvil tudi označevalnik besedil [16] in številni digitalni korpusi, kot je naprimer Gigafida [3]. Razčlenjevalnik nam omogoča, da besedam v povedih avtomatično pripišemo skladenjska razmerja. Osnovan je na razčlenjevalniku MSTParser [28], ki v usmerjenih grafih išče minimalno vpeto drevo. Razčlenjevalnik za skladenjsko razčlenjevanje uporablja sistem odvisnostnih drevesnic, ki je bil razvit v projektu Jezikoslovno označevanje slovenščine [11]. Vsaka poved je tako predstavljena v drevesni strukturi, kjer so povezave odvisnih besed predstavljene z enim od desetih tipov, opisanih v tabeli 3.1.

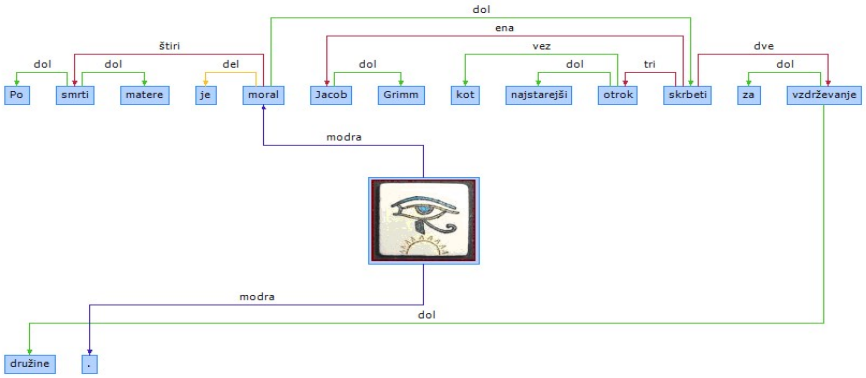
Tabela 3.1: Tipi povezav pri razčljenjevanju besedila.

| tip povezave | kaj povezuje |
|---|---|
| **dol** | jedro in določilo besednih zvez |
| **del** | deli zloženega povedka |
| **prir** | jedra v prirednih zvezah znotraj stavka |
| **vez** | besede ali ločila v vezniški vlogi |
| **skup** | nepolnopomenske besede, ki imajo zelo močno tendenco po sopojavljanju |
| **ena** | osebek stavka |
| **dve** | predmet stavka |
| **tri** | prislovno določilo lastnosti |
| **štiri** | ostala prislovna določila |
| **modra** | hierarhično najvišje pojavnice, skladenjsko manj predvidljive in oddaljene strukture, vrinki, ločila. |

Program lahko sprejme tekstovne datoteke ali besedila v formatu XML-TEI [5]. Za vsak stavek v vhodni datoteki besedam v stavku določi lemo, oblikoslovno obliko ter odvisnosti med besedami. Oblikoslovno obliko razčlenjevalnik označi z označevalnikom za slovenski jezik [16]. Lema je osnovna oblika besede. Pri oblikoslovnem označevanju vsaki besedi določimo besedno vrsto in njene značilnosti, npr. spol, sklon in število. Natančnost določanja kategorije, spola, sklona in števila naj bi bila 91,34%, natančnost določanja besedne vrste pa 98,30%. Pri tem naj bi bila natančnost določanja osnovne oblike besed ob upoštevanju velike začetnice 97,88% ter 98,55% ob neupoštevanju velike začetnice. Odvisnosti med besedami so predstavljene s povezavami, opisanimi v tabeli 3.1. Slika 3.1 prikazuje primer vizualno predstavljenega razčlenjenega stavka, na sliki 3.2 je primer razčlenjenega stavka v formatu XML-TEI. Razčlenjevalnik je dostopen na portalu slovenscina.eu [37] pod licenco Apache License v2.0 [42].

## 3.2 Algoritem P-PR za rangiranje vozlišč v grafu

Algoritem P-PR (personalizirani PageRank) za rangiranje vozlič v grafu je različica algoritma PageRank (PR) [33]. PR se uporablja za izračunavanje




Slika 3.1: Vizualna razčlenitev stavka

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <text>
    <body>
      <p xml:id="0">
        <s xml:id="0.0">
          <w lemma="po" msd="Dm" xml:id="0.0.1">Po</w>
          <S />
          <w lemma="smrt" msd="Sozem" xml:id="0.0.2">smrti</w>
          <S />
          <w lemma="mati" msd="Sozer" xml:id="0.0.3">matere</w>
          <S />
          <w lemma="biti" msd="Gp-ste-n" xml:id="0.0.4">je</w>
          <S />
          <w lemma="morati" msd="Ggnd-em" xml:id="0.0.5">moral</w>
          <S />
          <w lemma="Jacob" msd="Slmei" xml:id="0.0.6">Jacob</w>
          <S />
          <w lemma="Grimm" msd="Slmei" xml:id="0.0.7">Grimm</w>
          <S />
          <w lemma="kot" msd="Vd" xml:id="0.0.8">kot</w>
          <S />
          <w lemma="star" msd="Ppsmeid" xml:id="0.0.9">najstarejši</w>
          <S />
          <w lemma="otrok" msd="Somei" xml:id="0.0.10">otrok</w>
          <S />
          <w lemma="skrbeti" msd="Ggnn" xml:id="0.0.11">skrbeti</w>
          <S />
          <w lemma="za" msd="Dt" xml:id="0.0.12">za</w>
          <S />
          <w lemma="vzdrževanje" msd="Soset" xml:id="0.0.13">vzdrževanje</w>
          <S />
          <w lemma="družina" msd="Sozer" xml:id="0.0.14">družine</w>
          <c xml:id="0.0.15">.</c>
          <links>
            <link afun="dol" dep="0.0.1" from="0.0.2" />
            <link afun="štiri" dep="0.0.2" from="0.0.5" />
            <link afun="dol" dep="0.0.3" from="0.0.2" />
            <link afun="del" dep="0.0.4" from="0.0.5" />
            <link afun="modra" dep="0.0.5" from="0.0.0" />
            <link afun="ena" dep="0.0.6" from="0.0.11" />
            <link afun="dol" dep="0.0.7" from="0.0.6" />
            <link afun="vez" dep="0.0.8" from="0.0.10" />
            <link afun="dol" dep="0.0.9" from="0.0.10" />
            <link afun="tri" dep="0.0.10" from="0.0.11" />
            <link afun="dol" dep="0.0.11" from="0.0.5" />
            <link afun="dol" dep="0.0.12" from="0.0.13" />
            <link afun="dve" dep="0.0.13" from="0.0.11" />
            <link afun="dol" dep="0.0.14" from="0.0.13" />
            <link afun="modra" dep="0.0.15" from="0.0.0" />
          </links>
        </s>
      </p>
    </body>
  </text>
</TEI>
```

Slika 3.2: Razčlenitev stavka v XML-TEI formatu

pomembnosti spletnih strani. Spletna stran je pomembna, če se nanjo povezuje mnogo drugih spletnih strani. Pri tem se upošteva, kako pomembna je spletna stran, ki vodi do naše spletne strani, ter na koliko različnih spletnih strani kaže. Vsaka spletna stran predstavlja vozlišče v grafu. Algoritem je zamišljen tako, da imamo naključnega sprehajalca, ki se sprehaja od vozlišča do vozlišča. V vsakem vozlišču izračuna PR vrednost, nato se odloči, ali bo skočil na poljubno vozlišče v grafu ali nadaljeval v enem od naslednjih vozlišč. Priporočena vrednost, da nadaljuje, je parameter (0.85). Razlika med algoritmoma P-PR in PR je, da P-PR skoči na eno od vozlišč, ki smo jih podali, ponavadi v začetno vozlišče. S to spremembo algoritem deluje hitreje, saj PR vrednosti hitreje konvergirajo. P-PR algoritem je iterativen. V začetnem koraku zgradimo vektor vozlišč ($r^{(0)}$), ki jim damo vrednost 1, če so začetna vozlišča, oziroma 0, če niso. Nato v vsakem koraku računamo po enačbi (3.1).

$$r^{(k+1)} = p(A^T r^{(k)}) + (1-p)r^{(0)} \tag{3.1}$$

Tu nam $r^{(k)}$ v enačbi predstavlja ocenjeno vrednost PR po $k$-tih iteracijah, $p$ predstavlja vrednost, da bo naključni sprehajalec nadaljeval svoj sprehod, $A$ pa predstavlja matriko povezav med vozlišči. Matrika $A$ mora biti normalizirana [21]. To pomeni, da se vse vrednosti v poljubni vrstici seštejejo v 1. Na koncu dobimo za vsako vozlišče PR vrednost. Večja kot je PR vrednost, bolj je vozlišče pomembno.

## 3.3 Word2vec

Predstavitev besed word2vec je leta 2013 objavil Google [29]. Vrednosti word2vec ne izračuna samo en algoritem, ampak več algoritmov, med katerimi lahko izbiramo. Algoritmi za učenje uporabljajo nevronsko mrežo z enim skritim nivojem. Nevronsko mrežo naučimo relacije med besedami ter njihovo sopojavljanje v besedilih. Pri učenju ustvarimo model iz besedil v korpusu. Iz besed se ustvari visoko-dimenzionalni prostor, kjer so besede

predstavljene kot vektorji, ki so v prostoru razporejeni tako, da so med seboj bližji tisti, ki se v učnih besedilih pogosteje pojavljajo skupaj. Word2Vec ima dve vrsti arhitekture modelov za učenje besed. Prva temelji na neskončnih vrečah besed in iz trenutnega konteksta predvidi novo besedo. Neskončna vreča besed je zbirka objektov, ki so sestavljeni iz n besed skupaj v dokumentih, iz katerih jo gradimo. Pri gradnji vreče moramo upoštevati vrstni red besed. V to vrečo lahko dodajamo vedno nove objekte. Druga arhitektura temelji na neskončnih preskočnih-gramih. Preskočni-grami so v osnovi podobni neskončnim vrečam besed, vendar ne upoštevajo vrstnega reda besed v dokumentih. Pri gradnji objektov preskočni-grami za vsako besedo poiščejo vse mogoče kombinacije, ki jo določena beseda tvori z ostalimi za največ n mest oddaljenimi besedami. Pri neskončnih preskončnih-gramih model vzame trenutno besedo ter predvidi besede, ki so v okolici vhodne besede. Modeli z neskončnimi vrečami besed so pri učenju hitrejši, medtem ko so preskončni-grami počasnejši, vendar naj bi bolje ocenili redkejše besede. Oba modela lahko učimo še z negativnim vzorčenjem ter mnogimi drugimi algoritmi. Negativno vzorčenje pohitri učenje. Pri učenju želimo maksimizirati podobnost med vektorji, ki so blizu med seboj. Pri tem ocenjujemo podobnost med trenutno besedo in ciljno besedo. Če ne uporabljamo negativnega vzorčenja, moramo izračunati vse podobnosti med konteksti, v katerih se beseda nahaja in ciljno besedo. To je lahko počasno, ker imajo besede številne kontekste. Z negativnim vzorčenjem izberemo naključno nekaj kontekstov, ki niso povezani z našo besedo. Po avtorjevih besedah za angleški jezik dobimo najboljše rezultate, če uporabimo algoritem preskočni-grami z negativnim vzorčenjem. V jeziku python je najbolj znana implementacija word2vec-a iz knjižnice gensim [35].

## 3.4 ccGigafida

ccGigafida [9, 3] je uravnotežen slovenski korpus, ki vsebuje slovenska besedila različnih vrst. Korpus je nastal iz večjega korpusa, poimenovega Gigafida

[3]. Gigafida in ccGigafida sta nastali v sklopu projekta Sporazumevanje v slovenskem jeziku (`http://www.slovenscina.eu/`). ccGigafida je velika približno 9% Gigafide. Korpus vsebuje 31.722 dokumentov iz časopisov, revij, knjižnih publikacij, spletnih besedil, prepisov parlamentnih govorov ter podobnih dokumentov. Dokumenti v korpusu so zapisani v formatu XML-TEI [5]. Vsak dokument vsebuje informacijo o viru, letu nastanka, vrsti besedila, naslovu in avtorju, če je le-ta znan. Korpus je tudi oblikoskladenjsko označen, kar pomeni, da je vsaki besedi v korpusu pripisana lema ter oblikoslovna oblika. Dostopen je na portalu slovenscina.eu [26].

## 3.5 WordNet in SloWNet

WordNet [30] je leksikalna podatkovna baza za angleški jezk v obliki grafa. V njej so glagoli, samostalniki in ostale besedne vrste zbrani v skupnih seznamih sopomenk, imenovanih synseti, ki so med seboj povezani z semantičnimi in leksikalnimi relacijami. Tako so synseti med seboj lahko povezani z relacijo nadpomenka, podpomenka, holonim, meronim in ostalimi semantični relacijami. WordNet vsebuje okoli 155.000 synsetov. Uporablja se pri iskanju pomenov besed, avtomatski klasifikaciji besedil, avtomatskem povzemanju, strojnem prevajanju in na mnogih drugih področjih procesiranja naravnega jezika.

Slovenska različica WordNet-a je SloWNet [12]. SloWNet ima enake lastnosti kot WordNet. Vsebuje 43.460 synsetov. Zadnja verzija sloWnet-a je dostopna na CLARIN.SI repozitoriju [13].

# Poglavje 4

# Povzemanje

Naš sistem za povzemanje iz dokumenta pridobi trojice v obliki osebek-povedek-predmet. Besede v trojicah obogatimo s podatki iz baze sloWnet [12]. Podobne besede združimo z word2vec modelom in zgradimo graf. Vozlišča v grafu ocenimo s P-PR algoritmom. Posamezno P-PR vrednost besede v trojicah utežimo. Med uteženimi trojicami izberemo najboljše, jih uredimo in oblikujemo v povzetek. V nadaljevanju poglavja sledi podrobnejši opis faz tega postopka.

## 4.1 Pridobivanje semantičnih trojk

Vsak dokument razčlenimo z razčlenjevalnikom besedil za slovenski jezik. Razčlenjevalnik za slovenski jezik je razvit v javi [37], zaradi tega smo morali napisati ovoj za python, ki ga je znal uporabljati. Za razčlenjevanje smo uporabili model, ki je bil naučen na korpusu ssj500k [22]. Korpus SSJ500k je sestavljen iz celotnega korpusa JOS100k [11] ter iz 400.000 besed iz korpusa JOS1M [10]. Vsaka beseda v korpusu je označena z osnovno obliko besede in oblikoskladenjsko oznako. Korpus vsebuje tudi podatke o imenskih entitetah in je ročno pregledan. Pri razčlenjevanju besedila smo dobili za vsako besedo njeno lemo ter oblikoslovno obliko. Razčlenjevalnik za vsak posamezi stavek določi relacije med besedami. Relacije in oznake besed pomagajo pri iskanju

osebkov, predmetov in povedkov v besedilu.

Prvi korak je, da v besedilu poiščemo vse povedke. Začnemo z iskanjem vseh besed, ki so označene kot glagol. Pri tem izločimo glagole, ki se nahajajo v vprašalnih stavkih. Za vsak najden glagol pogledamo, če je v relaciji z drugo besedo v stavku s tipom povezave dol ali del (tabela 3.1). Glagole in besede povezane z njim, združimo v povedek. Za povedke v istem stavku pogledamo, če vsebujejo kakšno skupno besedo. Dva povedka imata skupno besedo, ko se beseda nahaja na istem mestu v stavku. Če najdemo tak par, ju združimo v skupni povedek.

Za vse najdene povedke v besedilu moramo poiskati še osebke in predmete. Pri tem si pomagamo z relacijami, ki so označene s tipom ena ali dve (tabela 3.1). Za vsak povedek preverimo, če ima obe relaciji. Če povedek vsebuje obe relaciji, se lotimo iskanja osebkov. V nasprotnem primeru ga zavržemo, ker ne vsebuje podatkov, ki bi jih kasneje potrebovali pri ustvarjanju povzetkov. Osebki so povezani s povedkom z relacijo povezave tipa ena. Najprej za dani povedek poiščemo vse besede, ki so v taki relaciji z njim. Za najdene besede rekurzivno poiščemo še vse besede, ki so v relaciji z njimi. To storimo za vsako besedo, ki jo najdemo, dokler najdena beseda nima več nove relacije. Na koncu vse najdene besede uredimo po mestu v našem stavku. Tako pridobimo osebke, povezane s prej dobljenimi povedki. Nato s podobnim postopkom poiščemo predmete. Postopek iskanja predmetov se razlikuje le v tem, da namesto relacije tipa ena iščemo relacije tipa dve.

Vsak najden osebek, povedek in predmet predstavlja semantično trojico. Veliko trojic vsebuje osebne zaimke. Ker vsaka trojica predstavljala svojo celoto, je potrebno ugotoviti, komu osebni zaimek pripada. To težavo smo želeli rešiti z iskanjem poimenovanih entitet (angl. named entities). Uporabili smo model, ki je bil razvit in preizkušen v okviru dela Razpoznavanje imenskih entitet v slovenskem besedilu [39], vendar nam je označevalnik na testnem korpusu preslabo označeval, da bi ga lahko uporabili za naše namene. Tako smo osebne zaimke zamenjali s pomočjo hevristike. Po premisleku smo

ugotovili, da so osebni zaimki označeni z razčlenjevalnikom večinoma istega spola kot iskana beseda. To velja predvsem za osebne zaimke v ednini. Besede, ki so potencialni kandidati za zamenjavo, so označene kot samostalniki. Pravilni samostalnik, ki ga iščemo za zamenjavo osebnega zaimka, je velikokrat v njegovi bližini. Velikokrat je oddaljen za največ dva stavka. Ker se lahko zgodi, da je več besed v okolici istega spola, uporabimo pri ocenjevanju kandidatov še sklon, število ter oddaljenost besede od izbranega osebnega zaimka.

Sistem v celotnem besedilu poišče besede, ki so označene kot samostalnik. Za vsak samostalnik pogleda, s katerimi besedami je povezan. Najdene besede skupaj s samostalnikom predstavljajo samostalniško frazo. Ko imamo samostalniške fraze, pregledamo trojice, če vsebujejo besedo, ki je označena kot osebni zaimek. Za najdeno besedo pogledamo v seznam s samostalniškimi frazami in za vsako samostalniško frazo, ki se pojavi v besedilu pred osebnim zaimkom ali v istem stavku kot osebni zaimek, podamo oceno (algoritem 1). Ocenjujemo vsak samostalnik v naši samostalniški frazi. Pri samostalniku pogledamo, če je v isti osebi, številu in sklonu kot osebni zaimek. Če se samostalnik in zaimek ujemata v spolu, dodamo končni oceni 3 točke, 2 točki, če se ujemata v številu ter 1 točko, če se ujemata v sklonu. Vse pridobljene točke seštejemo ter utežimo z oddaljenostjo samostalnika od zaimka. Samostalnik, oddaljen za več kot dva stavka, utežimo z 0.1, sicer oddaljenost odštejemo od tri in utežimo pridobljene točke. To storimo za vsak samostalnik v samostalniški frazi. Končna ocena samostalniške fraze je maksimalna ocena med vsemi osebnimi zaimki v samostalniški frazi. Ko ocenimo vse samostalniške fraze, zamenjamo naš zaimek s frazo, ki ima najvišjo oceno.

Menimo da naš sistema v primerjavi s tistim v delu Razpoznavanje imenskih entitet v slovenskem besedilu [39] uspešnejše zamenja osebne zaimke. Pri tem izhajamo iz tega, da iskalnik imenskih entitet v prej omenjenem delu ne označi veliko entitet. S tem izgubimo veliko potencialnih osebkov in tako pri zamenjavi zaimkom z najbližjo označeno entiteto prepogosto storimo napako. Naša hevristika s pomočjo predznanja ima na voljo številne kandi-

**Algoritem 1** Pseudokoda za zamenjavo osebnih zaimkov

```
procedure ZamenjavaOsebnihZaimkov(seznamTrojic, samostalniškeFraze)
    for beseda in seznamTrojic do                    ▷ Vsaka beseda v trojicah
        if tip(beseda) == „osebniZaimek“ then
            maksimalnaOcena ← 0
            najboljšaFraza ← null
            for samostalniškaFraza in samostalniškeFraze do
                for samostalnik in samostalniškaFraza do
                    razdalja ←položaj(samostalnik) - položaj(beseda)
                    if razdalja ≥ 0 then
                        ocena ← 0
                        if spol(samostalnik) == spol(beseda) then
                            ocena ← ocena + 3
                        if število(samostalnik) == število(beseda) then
                            ocena ← ocena + 2
                        if sklon(samostalnik) == sklon(beseda) then
                            ocena ← ocena + 1
                        if razdalja ≥ 3 then
                            ocena ← ocena · 0.1
                        else
                            ocena ← ocena · (3 − razdalja)
                        if ocena > maksimalnaOcena then
                            maksimalnaOcena ← ocena
                            najboljšaFraza ← samostalniškaFraza
            if najboljšaFraza ≠ null then
                beseda ← najboljšaFraza
```

date za zamenjavo, med katerimi s pomočjo pravil določi najustreznejšega. Bolje deluje, ker predvidevamo, da so kandidati v bližini osebnega zaimka, kar upoštevamo pri iskanju.

Preden začnemo graditi graf, pogledamo, če se katera trojica podvoji. Podvojene trojice zavržemo. Pogledamo tudi, če imata kateri trojici v nekem stavku isti osebek in povedek ali povedek ter predmet. Take trojice združimo. Trojice, ki jih dobimo iz besedila v dodatku A.1, so prikazane na sliki 4.1.


Slika 4.1: Pridobljene in urejene trojice. Temno modro so obarvani osebki, oranžno povedki in zeleno predmeti

Na sliki 4.1 opazimo, da so nekatere trojice uporabne, npr. „*je kot višji inženir prevzel vodstvo razvojnega oddelka letalske tovarne Ufag - Ungarische Flugzeugfabrik AG*“. Pravilno določimo predvsem povedke. Številne trojice so pomanjkljive. Primer slabo definirane trojice je „*Dunaju je umrla mati*“, kjer dobi trojica zaradi izgubljenih besed nov pomen. Trojice so tudi časovno slabo označene. Potrebno bi bilo ugotoviti, kdaj se je trojica zgodila, ter zamenjati časovne zaimke. Nekatere so oblikoslovno slabo povezljive med seboj, npr. „*Stanko ni umrl oče*“.

## 4.2 Bogatanje besed in združevanje isto pomenskih besed

Zdaj imamo zbirko trojic, ki smo jih poiskali v besedilu. Naslednji korak je poiskati besede v trojicah, ki so med seboj podobne, ter jih združiti, saj vsaka beseda kasneje predstavlja vozlišče v grafu. Poiščemo tudi sopomenke besed, ki jih kasneje uporabimo pri generiranju povzetka.

Na začetku vsako trojico uredimo tako, da so besede v njej v takem vrstnem redu, kot se pojavijo v osnovnem stavku. Potem iz vsake trojice odstranimo odvečne besede (angl. stopwords). To so besede, ki se pogosto pojavljajo v besedilih ter ne dodajo informacije. Velikokrat delujejo kot mašila v besedilih. Ko odstranimo odvečne besede, besedam pripišemo sopomenke. To storimo s pomočjo SloWNet-a. V diplomskem delu smo uporabili starejšo verzijo SloWNet-a, ki se nahaja v python knjižnici NLTK [7], ki je namenjena obdelavi naravnega jezika. V knjižnici NLTK je implementirana knjižnica z različicami WordNeta v različnih jezikih [4]. Podrobnejši opis knjižnice ter seznam jezikov, ki so na voljo, je na strani projekta Open Multilingual Wordnet [4]. S SloWNetom vsaki besedi poiščemo najbolj ustrezni synset s pomočjo predznanja oblikoslovne oblike ter s pomočjo poenostavljenega algoritma Lesk [19]. Algoritem Lesk je namenjen iskanju pomena besede v besedilu. Lesk je eden od mnogih algoritmov, ki pomagajo pri iskanju pomena besed v nekem kontekstu. Prilagojen je za uporabo z WordNet-om in v našem primeru za SloWNet. Algoritem vzame vsako besedo iz stavka ter za vsak synset preveri opis, kako dobro se ujema z ostalimi opisi besed v stavku. Synset, ki ima največ skupnih besed z opisi ostalih besed, je izbran in predstavlja najverjetnejši pomen besede v stavku. Pri štetju skupnih besed ne upoštevamo odvečnih besed.

Pri iskanju primernega synseta preverimo, če je beseda v SloWNet-u (algoritem 2). Če je ni, vrnemo vektor, ki vsebuje iskano besedo, sicer preverimo, da so kandidati v isti oblikoslovni obliki, kot iskana beseda in ob tem zavržemo tiste, ki niso. Če najdemo en synset, ga uporabimo in v naš vek-

tor za to besedo dodamo osnovno besedo ter njene sopomenke, sicer iščemo najbolj primeren synset s pomočjo Lesk-a. Za vsak potencialen synset preverimo ujemanje opisa z opisi ostalih besed v stavku. Če pri opisih ne dobimo ujemanja, vzamemo za najbolj ustreznega prvega na seznamu, saj predstavlja pomen besede, ki se največkrat pojavi v besedilih. Naši novi besedi pripišemo sopomenke iz najustreznejšega synseta.

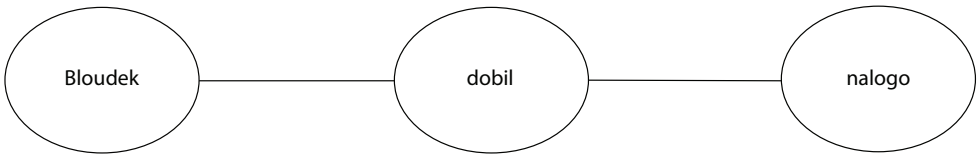
**Algoritem 2** Pseudokoda za bogatenje trojic

```
procedure BogatenjeTrojice(seznamTrojic, stavek)
    for beseda in seznamTrojic do   ▷ Zanka čez vse besede v trojicah
        if beseda in odvečneBesede then
            delete beseda                ▷ Če je beseda odvečna, jo izbrišemo
        else
            synsets ← SloWNet(beseda)          ▷ Poiščemo synsete besede
            if synsets == null then
                beseda ← [beseda]
            else
                maxUjemanje ← −1
                najboljšiSynset ← synsets[0]          ▷ Prvi na seznamu
                for synset in synsets do
                    if OblikoslovnaOblika(synset) ==
                            OblikoslovnaOblika(beseda) then
                        ujemanje ← poenostavljeniLesk(synset,
                            stavek(beseda))
                        if ujemanje > maxUjemanje then
                            maxUjemanje ← ujemanje
                            najboljšiSynset ← synset
                if maxUjemanje ≠ −1 then
                    beseda ← [beseda in vse besede v synset]
                else
                    beseda ← [beseda]
```

Ko besedam v trojicah pripišemo sopomenke, s pomočjo modela word2vec med seboj združimo besede, ki imajo isti pomen. Uporabljen model za word2vec je naučen s python knjižnico gensim na korpusu ccGigaFida. Model je dostopen na spletni strani virostatiq [34]. Pri učenju modela uporabimo unigrame, ki predstavljajo leme besed. Unigram je posamezna beseda v stavku. Word2vec uporabimo tako, da vsako besedo z vsemi sopomenkami v trenutni trojici uporabimo za primerjavo z ostalimi besedami in njihovimi sopomenkami v ostalih trojicah. Če primerjava besed vrne podobnost nad 0.75, pomeni, da sta si besedi med seboj podobni. Prag smo določili z eksperimentiranjem. Podobni besedi združimo tako, da oba para besed in njunih sopomenk združimo v isti vektor. Pri tem se besede v novem objektu ne podvajajo. V obeh trojčkih, ki vsebujeta podobni besedi, priredimo referenco na ta novo ustvarjeni vektor.

## 4.3 Gradnja grafa in ocenjevanje posameznih vozlišč

Ko besede obogatimo in združimo, iz njih zgradimo graf (algoritem 3). Vsaka beseda in njene sopomenke predstavljajo v grafu skupno vozlišče. Vozlišča so med seboj povezana po vrstnem redu besed v trojici. Tako so naprimer v trojici z osnovnimi besedami „Bloudek dobil nalogo“ vozlišča povezana med seboj, kot je prikazano na sliki 4.2.




Slika 4.2: Primer grafa iz trojice „Bloudek dobil nalogo“

Povezave smo poskusili utežiti z različnimi utežitvami. Najprej smo poskusili utežiti vse povezave z enako utežjo. Ta vrsta uteževanja ni nobene

povezave obravnavala kot pomembnejše od ostalih. Nato smo poskusili povezave utežiti s številom pojavljanj obeh osnovnih lem skupaj (dvogrami) v izbranem besedilu. Če se dvojica ni nikoli pojavila skupaj v besedilu, smo ji pripisali utež ena. Pri zadnji vrsti utežavanja smo za vsako lemo besede v vozliščih prešteli, kolikokrat se pojavi v osnovnem dokumentu. Utež nam predstavlja seštevek pojavljanj obeh vozlišč, ki sta povezani med seboj.

**Algoritem 3** Pseudokoda za gradnjo grafa

```
procedure ZgradiStavek(seznamTrojic, vrstaUtezi)
    grafPovezave ← []
    for trojica in seznamTrojic do                    ▷ Zanka čez trojice
        zacetnoVozlisce ← null
        for beseda in trojica do                      ▷ Zanka čez besede v trojici
            if zacetnoVozlisce ≠ null then
                if vrstaUtezi == „enako“ then
                    utez ← 1
                if vrstaUtezi == „dvogram“ then
                    if dvogram(zacetnoVozlisce, beseda) in vsiDvogrami
                            then
                        utez ← število(dvogram(zacetnoVozlisce, beseda))
                    else
                        utez ← 1
                if vrstaUtezi == „pojavitev“ then
                    utez ← število(zacetnoVozlisce) + število(beseda)
                grafPovezave.add([zacetnoVozlisce, beseda, utez])
                zacetnoVozlisce ← beseda
            else
                zacetnoVozlisce ← beseda
    graf ← zgradiGraf(grafPovezave)                   ▷ Zgradimo graf
    graf ← StohastičnaNormalizacija(graf)             ▷ Normaliziramo graf
    ocenaVozlišč ← P-PR(graf)                         ▷ Ocenimo vozlišča grafa z P-PR
```

Zgrajen graf je le šibko povezan in razpade na več manjših podgrafov.

Ocenimo, katera vozlišča so najpomembnejša v grafu. To storimo z algoritmom P-PR, pred tem pa graf še normaliziramo. Normalizirali smo ga z stohastično normalizacijo. Na normaliziranem grafu zaženemo P-PR, ki nam vrne oceno pomembnosti vozlišč. S pomočjo te ocene izberemo med trojicami tiste, ki so najbolj pomembne in jih damo v povzetek.

## 4.4 Generiranje povzetka

### 4.4.1 Gradnja slovarja z spreganimi besedami

Ker so besede v vozliščih grafa sestavljene iz lem, je pri gradnji povzetka potrebna uporaba slovarja spreganih besed. Slovar zgradimo s pomočjo korpusa ccGigaFida. Za vsako besedo v ccGigaFidi preverimo, da ne spada med odvečne besede. Ključ za iskanje po slovarju je lema besede. Vsaka lema ima shranjene besede, ki so v različnih sklonih, spolih ter številih. Tako novo besedo shranimo v slovar, če zanjo ni vnosa. Zgrajeni slovar na koncu shranimo v nerelacijsko bazo podatkov mongoDB [32]. Slovar spreganih besed je poenostavljena verzija SloLeksa [1].

### 4.4.2 Generiranje povzetka

Trojice iz besedila ocenimo. Vsaki besedi v trojici, ki ni odvečna, določimo vrednost, ki jo poda P-PR. Na začetku smo poskusili generirati povzetke samo z upoštevanjem te vrednosti, kasneje pa smo poskusili vrednost P-PR utežiti. Izkazalo se je, da z uteževanjem dobimo boljše povzetke. Za uteževanje besed v trojici smo preizkusili tri mere: TF-IDF, Okapi BM-25 ter pogostost besede v besedilu. Meri TF-IDF in Okapi BM-25 ocenjujeta pomembnost besede na nivoju posameznega odstavka v besedilu, Pogostost besede pa oceni pomembnost besede na nivoju celotnega članka.

Mera TF-IDF nam pove, kako pomembna je beseda v izbranem dokumentu v primerjavi s celotno zbirko dokumentov. Z njo lahko rangiramo dokumente po pomembnosti glede na poizvedovalno besedo. Uporablja se

predvsem v iskalnikih in pri klasifikaciji z vrečo besed. Vrednost, ki jo poda, se poveča, če se beseda večkrat pojavi v dokumentu ter redkeje v ostalih dokumentih. Izračuna se po enačbah,

$$TF(t,d) = \frac{f(t)}{|d|} \tag{4.1}$$

$$IDF(t,D) = log(\frac{N}{n}) \tag{4.2}$$

$$TF - IDF = TF(t,d) \cdot IDF(t,D) \tag{4.3}$$

Tu nam $f(t)$ predstavlja pogostost besede $t$ v dokumentu $d$ ter $|d|$ število besed v dokumentu $d$. $N$ predstavlja število vseh dokumentov v korpusu, $n$ pa število dokumentov, v katerem se iskana beseda pojavi vsaj enkrat. $D$ predstavlja vse dokumente v korpusu.

V našem sistemu smo mero nekoliko prilagodili. Posamezen dokument predstavlja en odstavek. Tako smo za pogostost besede izračunali, kolikokrat se beseda pojavi v odstavku. Za inverzno pogostost v dokumentih (IDF ) smo sešteli, kolikokrat se beseda pojavi v posameznemu odstavku ter izračunali vrednost po enačbi (4.2). Na koncu smo obe vrednosti zmnožili.

Okapi BM-25 je mera za rangiranje dokumentov glede na besedo. Izračuna se po enačbi

$$score\,(D,Q) = \sum_{i=1}^{n} IDF\,(q_i) \cdot \frac{f\,(q_i,D) \cdot (k_1+1)}{f\,(q_i,D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)} \tag{4.4}$$

pri čemer se $IDF(q_i)$ izračuna po enačbi (4.2) ter $f(q_i, D)$ po enačbi (4.1). $|D|$ je število besed v našem dokumentu, $avgdl$ predstavlja povprečno dolžino dokumentov. S parametroma $k_1$ in $b$ lahko prilagodimo delovanje mere; $k_1$ skalira pogostost pojavitve besede, $b$ pa nadzira normalizacijo dolžine dokumenta. Če je $k_1$ velik, v meri prevlada pogostost pojavitve besede, če je 0, pa mera predstavlja IDF. Priporočljiva vrednost za $k_1$ je med 1.2 in 2, priporočljiva vrednost za $b$ pa 0.75 [27].

Podobno, kot smo prilagodili Okapi BM-25, smo prilagodili tudi mero Okapi BM-25 torej dokument predstavlja en odstavek v besedilu. Sistem

uporabi za parameter $k_1$ vrednost 1.6 in za $b$ priporočeno vrednost 0.75. Za $k_1$ smo izbrano vrednost uporabili, ker je vmesna vrednost na priporočenem intervalu.

Meri Okapi BM-25 in TF-IDF smo uporabili za določanje pomembnosti besede na nivoju odstavka. Pri meri pogostosti besede v besedilu pa smo določili, kako pomembna je beseda na nivoju celotnega besedila. Izračuna se po enačbi (4.1)

**Algoritem 4** Pseudokoda za izbiro trojic

```
function IzberiTrojice
    izbraneTrojice ← []
    counter ← 0
    while counter ≤ log2(dolžinaBesdila) do
        najboljšaTrojica ← null
        maxVrednost ← 0
        for trojica in seznamTrojic do                ▷ Zanka čez trojice
            vrednostTrojice ← 0
            for beseda in trojica do    ▷ Zanka čez vektor besed v trojici
                if beseda not in izbraneTrojice then
                    utež ← utežitev(beseda)
                    vrednostP − PR ← P − PR(beseda) · utež
                    vrednostTrojice ← vrednostTrojice+vrednostP−PR
            if vrednostTrojice > maxVrednost then
                izbraneTrojice ← trojica
                najboljšaTrojica ← vrednostTrojice
        delete seznamTrojic[najboljšaTrojica] ▷ Izbrano trojico
                                                    izbrišemo
        izbraneTrojice.add(najboljšaTrojica)
        counter ← counter + 1
    return izbraneTrojice
```

Z vrednostmi mere, ki smo jo izračunali, utežimo P-PR vrednosti za vsako besedo v trojicah (algoritem 4). Vse ocene besed v trojici seštejemo, to pred-

stavlja oceno trojice. Trojice z najvišjimi ocenami vzamemo v končne povzetke. Ostale trojice na novo ocenimo, enako kot prvič, z razliko, da imajo besede, ki so že med izbranimi trojicami, P-PR vrednost enako nič. S tem onemogočimo, da bi se v končnem povzetku pojavljale redundantne informacije. Po izračunu ponovno izberemo najbolje ocenjeno trojico. Postopek ponavljamo, dokler ne izberemo dovolj trojic za povzetek. Število izbranih trojic je enako $log_2(|d|)$, kjer $|d|$ predstavlja dolžino osnovnega dokumenta. Vrednost smo dobili z eksperimentiranjem.

Ko imamo zbrane vse trojice za končni povzetek, jih uredimo po mestu, kjer so se nahajale v osnovnem dokumentu. Vsako trojico moramo urediti v stavek, saj so trenutno v njih samo vektorji lem in njihovih sopomenk (algoritem 5). Pri generiranju stavka uporabimo podatek, kako je originalni stavek izgledal in v kakšni oblikoslovni obliki so bile besede. To uporabimo pri generiranju stavka tako, da so besede v trojici oblikoslovno v enaki obliki kot v izvirnem besedilu. Izbrati moramo še prave besede oziroma sopomenke iz posameznih vektorjev. To storimo z word2vec modelom. Besede v nekem vektorju besed oziroma njihovih sopomenk primerjamo z ostalimi vektorji besed v tej trojici. V stavek vzamemo besedo, ki da najvišjo vrednost. Ker je beseda v obliki leme, za pravo obliko pogledamo v slovar spreganih besed. To ponovimo za vse besede v trojici ter za vse trojice, ki smo jih izbrali v povzetek. S tem postopek generiranja povzetka zaključimo.

**Algoritem 5** Pseudokoda za generiranje povzetka

```
function GenerirajPovzetek(izbraneTrojice)
    povzetek ← []
    izbraneTrojice ← urediPoVrstnemRedu(izbraneTrojice)
    for trojica in izbraneTrojice do                    ▷ Zanka čez trojice
        for beseda in trojica do      ▷ Zanka čez vektorje besed v trojici
            maxVrednost ← 0
            izbranaBeseda ← null
            stavek ← []
            for lema in beseda do
                vrednost ← word2vec(lema, ostaleBesedeVStavku)
                if vrednost > maxVrednost then
                    maxVrednost ← vrednost
                    izbranaBeseda ← lema
            stavek.add(slovar(izbranaBeseda)) ▷ Izbrano besedo uredimo
                                                v obliko originalne besede
        stavek ← uredi(stavek) ▷ Dodamo ločila in odvečne besede
                                   iz originalne trojice
        povzetek.add(stavek)
    return povzetek
```

# Poglavje 5

# Evalvacija

V poglavju je predstavljen korpus, ki smo ga zgradili iz člankov na Wikipediji in ki ga uporabimo za testiranje sistema. Predstavimo mere, s katerimi smo ocenili ustvarjene povzetke, ter predstavimo rezultate.

## 5.1 Testni korpus

Za testiranje sistema smo uporabili korpus, ki ga sestavljajo članki s slovenske Wikipedije. Naslove člankov smo pridobili iz baze DBpedia [24]. Z naslovi pa smo pridobili članke iz Wikipedije. Pri tem smo si pomagali s python knjižnico wikipedia [43]. Za vsak članek smo začetni del shranili kot povzetek, ločeno od glavne vsebine. Ta povzetek je služil za primerjavo z generiranim povzetkom. Vsebino z glavnega članka smo nekoliko prečistili, preden smo ga uporabili. Iz besedila smo izbrisali vse reference in vire, ki vsebinsko niso pomembni, zbrisali smo odvečne presledke in znake, ki so označevali enačbe ali slike. Ko smo besedilo očistili, smo članek shranili le v primeru, da je vseboval vsaj 8000 znakov v jedru besedila in vsaj 100 znakov v povzetku. Tako izbrane članke smo shranili v tekstovne datoteke. Pri tem smo pazili, da ni bilo podvajanja. Testni korpus je sestavljen iz 777 dokumentov z različnih področij.

## 5.2 Mere za evalvacijo

Za evalvacijo uspešnosti povzemanja uporabimo meri ROUGE-N in ROUGE-S [25].

ROUGE-N je mera, ki pove, v koliko n-gramih se ujemata naš povzetek in povzetek, s katerim ga primerjamo (referenčni povzetek). $N$ ppredstavlja število besed v n-gramu. Mera se izračuna po enačbi,

$$M_{gram_n} = \sum_{S\epsilon\{R\}} \sum_{gram_n\epsilon S} Count_{match}(gram_n) \tag{5.1}$$

$$N_{gram_n} = \sum_{S\epsilon\{R\}} \sum_{gram_n\epsilon S} Count(gram_n) \tag{5.2}$$

$$ROUGE - N = \frac{M_{gram_n}}{N_{gram_n}} \tag{5.3}$$

kjer $M_{gram_n}$ predstavlja število n-gramov ki se ujemajo z vsemi referenčnimi povzetki, $N_{gram_n}$ pove število n-gramov v referenčnih povzetkih, $R$ predstavlja zbirko vseh referenčnih povzetkov, $S$ en referenčni povzetek, $count(gram_n)$ število n-gramov v referenčnem povzetku, $count_{match}(gram_n)$ pa označuje, koliko n-gramov se ujema v našem povzetku z referenčnim povzetkom. Pri testiranju smo generirane povzetke testirali z ROUGE-1, ROUGE-2 in ROUGE-3.

ROUGE-S namesto n-gramov uporablja preskočne-grame. Preskočnigrami so vse možne kombinacije besed dolžine $n$ v besedilu. Podobnost preskočnih-gramov dolžine dve izračunamo po enačbah,

$$R_{skip2} = \frac{SKIP2(X,Y)}{C(m,2)} \tag{5.4}$$

$$P_{skip2} = \frac{SKIP2(X,Y)}{C(n,2)} \tag{5.5}$$

$$F_{skip2} = \frac{(1+\beta^2)R_{skip2}P_{skip2}}{R_{skip2} + \beta^2 P_{skip2}} \tag{5.6}$$

kjer $SKIP2(X,Y)$ predstavlja število skupnih preskočnih-gramov med našim in referenčnim povzetkom, $m$ predstavlja dolžino referenčnega povzetka, $n$ pa

dolžino našega povzetka. Funkcija $C(m, 2)$ oziroma $C(n, 2)$ nam pove, koliko kombinacij lahko generiramo s preskočnimi-grami dolžine dva. $P_{skip2}$ predstavlja mero točnost, $R_{skip2}$ pa predstavlja priklic. Točnost pove delež pravilno pozitivnih napovedanih besed izmed vseh možnosti. V našem primeru torej predstavlja delež pravilno napovedanih dvogramov med vsemi dvogrami v povzetku. Priklic pove delež pravilno pozitivnih napovedanih besed izmed vseh dejansko pozitivnih. V našem primeru pove, kolikšen delež dvogramov se ujema med referenčnimi povzetki in našim povzetkom izmed vseh dvogramov v referenčnih povzetkih. Za končno oceno podobnosti izračunamo F1-mero. V enačbi (5.6), $\beta$ kontrolira relativno pomembnost $R_{skip2}$ in $P_{skip2}$. Med seboj smo primerjali preskočne-grame dolžine dve.

Za boljši občutek, kako uspešni so rezultati, smo tri članke iz različnih področij povzeli ročno. Članke je povzelo pet ljudi. Ročne povzetke smo primerjali s testnimi povzetki.

Med testiranjem smo preizkusili različne uteži za uteževanje povezav v grafu ter različno uteževanje vrednosti besed, ki nam jo poda P-PR. Za uteževanje povezav grafov smo uporabili uteži število pojavitev dvogramov, število pojavitve besede v besedilu ter enakomerno uteževanje. Pri uteževanju P-PR vrednosti smo uporabili nevtralno utež ena ter uteži, ki so jih podale mere TF-IDF, Okapi BM-25 ter frekvenca besede v dokumentu. Postopki za uteževanje P-PR vrednosti in povezav so opisani v razdelku 4.4.2.

## 5.3 Rezultati

Rezultate z enakomernim uteževanjem P-PR vrednosti prikažemo v tabelah 5.1 in 5.2. Iz tabel lahko razberemo, da dobimo najboljše rezultate, če povezave grafov utežimo s seštevkom pojavitve obeh besed, ki ju povezujeta vozlišči. S to utežitvijo dobimo najboljše rezultate pri testih za ROUGE-1, ROUGE-2 in ROUGE-S. Le pri ROUGE-3 dobimo boljše rezultate pri uteževanju grafa z dvogrami ter z enakomernim uteževanjem, vendar ni velike razlike med različnimi uteževanji. Drugo najboljše uteževanje povezav v

Tabela 5.1: Povprečne vrednosti z enakomerno utežitvijo P-PR.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| enakomerno | 0.093 | 0.0055 | 0.0010 | 0.0049 |
| dvogrami | 0.09397 | 0.0057 | 0.0010 | 0.0049 |
| seštevek pojavitve | 0.0974 | 0.0062 | 0.0007 | 0.0051 |

Tabela 5.2: Standardni odklon z enakomerno utežitvijo P-PR.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| enakomerno | 0.0705 | 0.0139 | 0.008 | 0.0077 |
| dvogrami | 0.0716 | 0.0137 | 0.0074 | 0.0077 |
| seštevek pojavitve | 0.07 | 0.0125 | 0.0044 | 0.0068 |

grafih pri tem načinu je uteževanje s številom dvogramov.

Naslednja mera, s katero smo preizkusili utežavanje P-PR vrednosti, je TF-IDF. Rezultati so v tabelah 5.3 in 5.4. Pri tej meri se rezultati med seboj le malo razlikujejo, vendar je bilo ponovno za malenkost boljše uteževanje grafov s seštevkom pojavitve, ki ima boljši rezultat pri ROUGE-2 in ROUGE-3. Drugo najboljše uteževanje je z pojavitvijo dvogramov. V primerjavi z enakomernim uteževanjem P-PR vrednosti dobimo s TF-IDF uteževajem nekoliko boljše rezultate.

Tabela 5.3: Povprečne vrednosti za TF-IDF.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| TF-IDF&enakomerno | 0.1027 | 0.0071 | 0.0012 | 0.0059 |
| TF-IDF&dvogrami | 0.1021 | 0.0072 | 0.0011 | 0.0059 |
| TF-IDF&seštevek pojavitve | 0.1022 | 0.0074 | 0.0013 | 0.0059 |

Tabela 5.4: Standardni odklon za TF-IDF.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| TF-IDF&enakomerno | 0.0758 | 0.0168 | 0.0083 | 0.0074 |
| TF-IDF&dvogrami | 0.0716 | 0.0161 | 0.0078 | 0.007 |
| TF-IDF&seštevek pojavitve | 0.0734 | 0.0164 | 0.0076 | 0.0074 |

Ker se je mera TF-IDF bolje odrezala od enakomernih uteži, smo za utežavanje preizkusili še Okapi BM-25 mero. Pri tem smo pridobili rezultate prikazane v tabelah 5.5 in 5.6. V tem primeru ne dobimo najboljšega rezultata s seštevkom pojavitve, ampak senakomerne uteži in dvogrami izkažejo kot boljše uteževanje. Med njima ni velike razlike, vendar smo z mero Okapi BM-25 pridobili nekoliko boljše rezultate kot s TF-IDF.

Tabela 5.5: Povprečne vrednosti za Okapi BM-25.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| Okapi BM-25&enakomerno | 0.1103 | 0.0086 | 0.0016 | 0.0068 |
| Okapi BM-25&dvogrami | 0.1109 | 0.0085 | 0.0015 | 0.0068 |
| Okapi BM-25&seštevek pojavitve | 0.1086 | 0.0084 | 0.0014 | 0.0066 |

Tabela 5.6: Standardni odklon za Okapi BM-25.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| Okapi BM-25&enakomerno | 0.0798 | 0.0193 | 0.01 | 0.0087 |
| Okapi BM-25&dvogrami | 0.0772 | 0.0184 | 0.0097 | 0.0087 |
| Okapi BM-25&seštevek pojavitve | 0.0767 | 0.018 | 0.0088 | 0.008 |

Za zaključek smo preizkusili še uteževanje P-PR vrednosti s frekvenco posamezne besede, rezultati so v tabelah 5.7 in 5.8. Rezultati različnih

uteževanj povezav grafov so podobni. Pri merah ROUGE-1 in ROUGE-2 nekoliko izstopa uteževanje s povezavami grafa. Nasprotno je pri merah ROUGE-3 in ROUGE-S, kjer je enakomerno uteževanje najslabše. Zaradi tega ocenimo, da je utežavenje z dvogrami najboljše, saj dobimo boljše rezultate pri merah, ki upoštevajo daljše nize. Med vsemi merami, ki smo jih uporabili za utežavanje P-PR vrednosti, se je najbolje odrezala ravno ta utežitev.

Tabela 5.7: Povprečne vrednosti za uteževanje s frekvenco.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| frekvenca&enakomerno | 0.1194 | 0.0103 | 0.00167 | 0.0077 |
| frekvenca&dvogrami | 0.1170 | 0.01 | 0.0018 | 0.0078 |
| frekvenca&seštevek pojavitve | 0.1164 | 0.0101 | 0.0018 | 0.0078 |

Tabela 5.8: Standardni odklon za uteževanje s frekvenco.

| uteževanje | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| frekvenca&enakomerno | 0.0852 | 0.0197 | 0.0082 | 0.0136 |
| frekvenca&dvogrami | 0.0863 | 0.0191 | 0.0083 | 0.0135 |
| frekvenca&seštevek pojavitve | 0.0897 | 0.0196 | 0.0081 | 0.0137 |

Za vsako od P-PR uteževanj grafov smo pogledali za najboljše uteževanje, kako dobre in berljive povzetke dobimo. Primerjali smo jih na povzetkih, ki jih dobimo iz besedil v dodatkih A.1 in A.3. Povzetki, ki smo jih dobili iz dodatka A.1 so v dodatku C.1, povzetki iz dodatka A.3 pa so v dodatku C.2.

Ko primerjamo povzetke ugotovimo, da so si precej podobni, večinoma se razlikujejo le v določenih stavkih. Vsebujejo številne slovnične napake. V nekaterih primerih so stavki nepovezani (dodatek C.1.1). Iz vseh lahko izvemo nekaj bistvenih podatkov, ki jih vsebuje osnovni dokument. Če primerjamo različne metode uteževanja, ugotovimo, da najboljše povzetke poda metoda,

ki utežuje graf s številom pojavitev dvogramov, P-PR vrednost pa s frekvenco pojavitve besede. Najslabše povzetke generira model brez uteževanja P-PR vrednosti.

Po izračunu ROUGE vrednosti in pregledu povzetkov pridemo do ugotovitve, da se je najboljše odrezal model, ki utežuje graf s številom pojavitev dvogramov, P-PR vrednost pa utežuje s frekvenco pojavitve besede. Rezultate te metode smo na koncu primerjali z ročnimi povzetki. Pet ljudi je povzelo besedila A.1, A.2 in A.3, rezultati, so predstavljeni v tabelah 5.9 in 5.10. Če te rezultate primerjamo z vrednostmi v tabeli 5.11, ugotovimo, da so povzetki, ki so jih ustvarili ljudje, precej boljši. Primer človeškega povzetka je dodatek D, kjer povzamemo besedilo v dodatku A.3. Hitro ugotovimo, tudi da so ljudje napisali precej daljše povzetke od našega sistema, kar vpliva na mero ROUGE-N, saj je verjetnost, da zadanemo besede v referenčnem povzetku, precej večja. Dolžina povzetkov manj vpliva na oceno ROUGE-S, vendar so tudi pri ocenjevanju s to mero ročni povzetki boljši. Boljša je tudi berljivost ročnih povzetkov in slovnična pravilnost. Če primerjamo ročne in generirane povzetke, ugotovimo, da generirani vsebujejo precej bolj strnjene in jedrnate stavke, predvsem generirani povzetki ne vsebujejo mašil in ustavitvenih besed.

Tabela 5.9: Povprečna ocena za povzetke, ki so jih ustvarili ljudje.

| besedilo | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| bloudek | 0.2051 | 0.0133 | 0.0 | 0.004 |
| egipčanski hieroglifi | 0.6204 | 0.04 | 0.0109 | 0.0053 |
| vanilija | 0.6667 | 0.0694 | 0.0098 | 0.0316 |
| **povprečje** | **0.4974** | **0.1227** | **0.0069** | **0.0136** |

Tabela 5.10: Standardni odklon za povzetke, ki so jih ustvarili ljudje.

| besedilo | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| bloudek | 0.14561 | 0.0231 | 0.0 | 0.0051 |
| egipčanski hieroglifi | 0.2694 | 0.08 | 0.0217 | 0.0033 |
| vanilija | 0.257 | 0.0233 | 0.0113 | 0.0023 |

Tabela 5.11: Povprečna ocena za povzetke, ustvarjene z uteževanjem povezav v grafu z dvogrami in P-PR vrednosti s frekvenco besede.

| besedilo | ROUGE-1 | ROUGE-2 | ROUGE-3 | ROUGE-S |
|---|---|---|---|---|
| bloudek | 0.1154 | 0.0 | 0.0 | 0.0 |
| egipčanski hieroglifi | 0.2592 | 0.0 | 0.0 | 0.0101 |
| vanilija | 0.2456 | 0.0 | 0.0 | 0.0243 |
| **povprečje** | **0.2067** | **0.0** | **0.0** | **0.0115** |

# Poglavje 6

# Sklepne ugotovitve

V diplomskem delu smo razvili abstraktivni povzemalnik slovenskih dokumentov pri tem smo osnovni dokument s pomočjo razčlenjevalnika predstavili kot trojice osebek-povedek-predmet. Trojice smo obogatili s pomočjo SloWNet-a. Podobne obogatene besede smo združili. Podobnost besed smo določili z word2vec modelom. Iz dobljenih besed smo zgradili graf. Za uteži med vozlišči smo preizkusili nekaj različnih tehnik uteževanja. Najbolje se je izkazala metoda uteževanja z dvogrami. Vsako vozlišče v grafu smo ocenili s pomočjo algoritma P-PR, zatem smo vsako besedo v trojici utežili z različnimi merami. Pri tem uteževanju se je najbolj izkazala frekvenca besede v osnovnem dokumentu. Ko smo izbrali primerne trojice, smo z njimi zgradili povzetek. Dobljeni povzetki so kratki in jedrnati. Izkazalo se je, da vsebujejo številne slovnične napake, nekateri pa so slabo formulirani, prav tako večkrat ne delujejo povezano. Iz njih lahko izvemo bistvene informacije, a sistem vseeno sestavi bistveno slabše povzetke, kot jih ročno pripravi človek.

Problem našega sistema je, da na začetku zavržemo številne besede in trojke in s tem številne informacije. Zaradi tega bi sistem imel težave pri povzemanju zelo kratkih dokumentov. Pri zavrženih trojicah, kjer manjka osebek, bi lahko uporabili prepoznavanje imenskih entitet in s tem poiskali najbolj primeren osebek. Z boljšim prepoznavanjem imenskih entitet bi lahko

izboljšali zamenjavo osebnih zaimkov. Prav tako bi bilo zanimivo preizkusiti še drugačen način povezovanja grafov. Pri izbiranju najboljših trojčkov bi bilo zanimivo preizkusiti, če lahko z metodami strojnega učenja bolje izbiramo med kandidati. Za generiranje stavkov bi bilo potrebno beležiti čas dogajanja trojic. Tako bi lahko pri povzemanju razvrstili trojice v časovno zaporedje. Potrebno je izboljšati formulacijo stavkov. Pri generiranju bi si lahko pomagali z vnaprej pripravljenimi vzorci stavkov. Namesto slovarja spreganih besed bi bilo smiselno uporabiti SloLeks. Naš sistem je smiselno preizkusiti še na drugih korpusih dokumentov, ki niso sestavljeni le iz člankov Wikipedije.

# Literatura


[1] Špela Arhar. *Učni korpus SSJ in leksikon besednih oblik za slovenščino.* Slavistično društvo Slovenije, 2009.

[2] Regina Barzilay, Kathleen R McKeown in Michael Elhadad. Information fusion in the context of multi-document summarization. V: *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, str. 550–557. Association for Computational Linguistics, 1999.

[3] Nataša Logar Berginc, Miha Grčar, Marko Brakus, Tomaž Erjavec, Špela Arhar Holdt, Simon Krek in Iztok Kosem. *Korpusi slovenskega jezika Gigafida, KRES, ccGigafida in ccKRES: gradnja, vsebina, uporaba.* Trojina, zavod za uporabno slovenistiko, 2012.

[4] Francis Bond in Ryan Foster. Linking and extending an open multilingual wordnet. V: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, str 1352–1362, Sofia, 2013. Association for Computational Linguistics.

[5] TEI Consortium. TEI P5: Guidelines for electronic text encoding and interchange. `http://www.tei-c.org/Guidelines/P5/`. TEI Consortium. (Obiskano: 20.8.2016).

[6] Kaja Dobrovoljc, Simon Krek in Jan Rupnik. Skladenjski razčlenjevalnik za slovenščino. V: *Tomaž Erjavec, Jerneja Žganec Gros (ur.): Zbornik*

*Osme konference Jezikovne tehnologije*, str. 42–47. Institut Jožef Stefan, 2012.

[7] Natural Language Toolkit — NLTK 3.0 documentation. `http://www.nltk.org/`. (Obiskano: 1.8.2016).

[8] Michael Elhadad. *Using argumentation to control lexical choice: a functional unification implementation*. Doktorsko delo, Columbia University, 1993.

[9] Tomaž Erjavec in Nataša Logar Berginc. Referenčni korpusi slovenskega jezika (cc) Gigafida in (cc) KRES. V: *Tomaž Erjavec, Jerneja Žganec Gros (ur.): Zbornik Osme konference Jezikovne tehnologije. Ljubljana: Institut Jožef Stefan*, str. 57–62, 2012.

[10] Tomaž Erjavec in Simon Krek. Training corpus JOS1M 1.1. `http://hdl.handle.net/11356/1037`, 2010. Slovenian language resource repository CLARIN.SI.

[11] Tomaž Erjavec, Darja Fišer, Simon Krek in Nina Ledinek. The JOS Linguistically Tagged Corpus of Slovene. V: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010. European Language Resources Association (ELRA).

[12] Darja Fišer. *Izdelava slovenskega semantičnega leksikona z uporabo eno- in večjezičnih jezikovnih virov*. Doktorsko delo, Faculty of Arts, Ljubljana, Slovenia, 2009.

[13] Darja Fišer. Semantic lexicon of Slovene sloWNet 3.1. `http://hdl.handle.net/11356/1026`, 2015. Slovenian language resource repository CLARIN.SI.

[14] Pierre-Etienne Genest in Guy Lapalme. Framework for abstractive summarization using text-to-text generation. V: *Proceedings of the Wor-*

*kshop on Monolingual Text-To-Text Generation*, str. 64–73. Association for Computational Linguistics, 2011.

[15] Pierre-Etienne Genest in Guy Lapalme. Fully abstractive approach to guided summarization. V: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, str. 354–358. Association for Computational Linguistics, 2012.

[16] Miha Grčar, Simon Krek in Kaja Dobrovoljc. Obeliks: statisticni oblikoskladenjski označevalnik in lematizator za slovenski jezik. V: *Zbornik Osme konference Jezikovne tehnologije, Ljubljana, Slovenia*. Institut Jožef Stefan, 2012.

[17] Wikipedija: Hieroglif. `https://sl.wikipedia.org/wiki/Hieroglif`. (Obiskano: 20.8.2016).

[18] Atif Khan in Naomie Salim. A review on abstractive summarization methods. *Journal of Theoretical and Applied Information Technology*, 59(1):2005, 2014.

[19] Adam Kilgarriff in Joseph Rosenzweig. English senseval: Report and results. V: *Proceedings of the 2nd Conference on Language Resources and Evaluation (LREC)*. Athens, Greece, 2000.

[20] Richard I Kittredge in Igor Mel'čuk. Towards a Computable Model of Meaning-Text Relations Within a Natural Sublanguage. V: *Proceedings of the Eighth International Joint Conference on Artificial Intelligence (IJCAI-83)*, str. 657–659, 1983.

[21] Jan Kralj, Anita Valmarska, Marko Robnik-Šikonja in Nada Lavrač. Mining text enriched heterogeneous citation networks. V: *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*, str. 672–683. Springer, 2015.

[22] Simon Krek, Tomaž Erjavec, Kaja Dobrovoljc, Sara Može, Nina Ledinek in Nanika Holz. Training corpus SSJ500k 1.3. http://hdl.handle.net/11356/1029, 2013. Slovenian language resource repository CLARIN.SI.

[23] Chang-Shing Lee, Zhi-Wei Jian in Lin-Kai Huang. A fuzzy ontology and its application to news summarization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(5):859–880, 2005.

[24] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer in Chris Bizer. DBpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal*, 2014.

[25] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. V: *Text summarization branches out: Proceedings of the ACL-04 workshop*, zvezek 8. Barcelona, Spain, 2004.

[26] Nataša Logar, Tomaž Erjavec, Simon Krek, Miha Grčar in Peter Holozan. Written corpus ccGigafida 1.0, 2013. Slovenian language resource repository CLARIN.SI. http://hdl.handle.net/11356/1035.

[27] Christopher D. Manning, Prabhakar Raghavan in Hinrich Schütze. Introduction to information retrieval. str 233. Cambridge University Press, 2009.

[28] Ryan McDonald, Kevin Lerman in Fernando Pereira. Multilingual dependency analysis with a two-stage discriminative parser. V: *Proceedings of the Tenth Conference on Computational Natural Language Learning*, str. 216–220. Association for Computational Linguistics, 2006.

[29] Tomas Mikolov, Ilya Sutskever in Quoc Le. Learning the meaning behind words. https://google-opensource.blogspot.si/2013/08/learning-meaning-behind-words.html. (Obiskano: 20.8.2016).

[30] AG Miller. Wordnet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[31] Ibrahim F Moawad in Mohammadreza Aref. Semantic graph reduction approach for abstractive Text Summarization. V: *Seventh International Conference on Computer Engineering & Systems (ICCES), 2012*, str. 132–138. IEEE, 2012.

[32] MongoDB. `https://www.mongodb.com/`. (Obiskano: 7.8.2016).

[33] Lawrence Page, Sergey Brin, Rajeev Motwani in Terry Winograd. The PageRank citation ranking: bringing order to the web. Technical report, Stanford InfoLab. 1999.

[34] Marko Plahuta. Virostatiq. `http://virostatiq.com/downloads/`. (Obiskano: 13.7.2016).

[35] Radim Řehůřek in Petr Sojka. Software Framework for Topic Modelling with Large Corpora. V: *Proceedings of the Language Resources and Evaluation (LREC) 2010 Workshop on New Challenges for NLP Frameworks*, str. 45–50, Valletta, Malta, May 2010. European Language Resources Association (ELRA). `http://is.muni.cz/publication/884893/en`.

[36] Jacques Robin. *Revision-based generation of Natural Language Summaries providing historical Background*. Doktorsko delo, Columbia University, 1994.

[37] Jan Rupnik, Janez Brank, Miha Grčar, Matjaž Juršič, Simon Krek in Igor Mozetič. Programska oprema za razčlenjevalnik. `http://razclenjevalnik.slovenscina.eu/Programska_oprema.aspx`. (Obiskano: 10.7.2016).

[38] Horacio Saggion in Guy Lapalme. Generating indicative-informative summaries with sumUM. *Computational linguistics*, 28(4):497–526, 2002.

[39] Tadej Štajner, Tomaž Erjavec in Simon Krek. Razpoznavanje imenskih entitet v slovenskem besedilu. V: *Proceedings of 15th Internation Multiconference on Information Society-Jezikovne Tehnologije*, 2012.

[40] Wikipedia: Stanko Bloudek. `https://en.wikipedia.org/wiki/Stanko_Bloudek`. (Obiskano: 20.8.2016).

[41] Wikipedija: Vanilja. `https://sl.wikipedia.org/wiki/Vanilja`. (Obiskano: 20.8.2016).

[42] Apache License Version 2.0. `http://www.apache.org/licenses/LICENSE-2.0`. (Obiskano: 26.8.2016).

[43] wikipedia 1.4.0 : Python Package Index. `https://pypi.python.org/pypi/wikipedia/`. (Obiskano: 1.8.2016).

# Dodatek A

# Originalna besedila

## A.1 Stanko Bloudek

(Vir: Wikipedia [40])

Zgodnje obdobje ==

Rodil se je v Idriji 11. februarja 1890, kot tretji otrok od skupno petih - imel je kar štiri sestre, po dve mlajši in dve starejši. Njegov oče Jaroslav je bil strojni inženir, doma iz Telč na Moravskem, zaposlen v idrijskem rudniku živega srebra, medtem, ko je bila mati Minka, rojena Lapajne, Slovenka. Leta 1894 se je cela družina preselila na Češko v Most (Brüx), kjer je oče dobil službo v rudniku rjavega premoga. Tu je mladi Stanko opravil osnovno šolo in štiri razrede klasične gimnazije, dokler mu leta 1904 zaradi kapi ni umrl oče. Sledila je selitev nazaj v Slovenijo, kjer je najprej živel pri stricu in opravil peti razred v Kranju, nato pa zadnje tri v Ljubljani na 1. državni gimnaziji. Zanimivo je, da je mladi Bloudek dosegal le povprečen učni uspeh - dober je bil pri telovadbi in risanju, medtem ko je prav matematiko in fiziko komaj izdeloval - ne ravno značilno za kasnejšega inženirja.

Vendar pa je tudi res, da je bil od tehnike takrat še zelo oddaljen, kajti Bloudkovi so se družili z bolj v umetnost usmerjenimi krogi, med drugimi je bratranec po materini strani Dolfe Lapajne študiral slikarstvo. Tako se je tudi sam Stanko navdušil nad slikanjem in že med dijaštvom obiskoval

zasebno slikarsko šolo pri slovenskem impresionistu Rihardu Jakopiču. Nato je, po maturi leta 1908 začel najprej s študijem slikarstva na likovni akademiji v Pragi, vendar je po enem letu presedlal na tehniško smer in se vpisal na visoko tehniško šolo, torej tako kot oče, za strojnika.

Leta 1910 mu je na Dunaju za rakom umrla še mati. Bloudek, ki se je že tako skromno preživljal in si je iskal številna priložnostna dela ob šolanju, se je tako soočil s hudo izgubo še drugega starša v svojem odraščanju. Resno življenjsko krizo je prebrodil s svojo predanostjo študiju in delu ter si nekako v tistem času našel usmeritev, ki je zaznamovala njegovo nadaljnjo pot - letalstvo.

== Letalstvo ==

Zelo zgodaj se je začel ukvarjati tudi s konstrukcijo letal. Že od leta 1906 je delal lastne letalske modele. V Pragi je naredil leta 1910 sprva jadralno, nato pa enokrilno motorno letalo, ki ga je poimenoval Racek - Galeb. Leta 1911 je izdelal dvokrilno motorno letalo z imenom Libela. V letih 1911 - 1918 je bil prvi slovenski poklicni konstruktor v letalski industriji. Deloval je v Leipzigu, na Dunaju, v Budimpešti in Trutnovu na Češkem.

Njegova prva služba v letalski industriji je bila pri Etrichu, podjetju, ki se je kasneje združilo z veliko bolj znanim Hansa-Brandenburg. Kot zanimivost velja omeniti, da je bil Bloudek že tretji Slovenec, ki je delal v tem podjetju, kajti v letih 1903- 1909 je bil tam najprej Mariborčan Franc Wels, nato pa še modelar Pavel Podgornik v 1909 in 1910. Tedaj je sodeloval pri izgradnji osnovnega Etrichovega letala Etrich taube, ki je bilo takrat eno najmodernejših nasploh. Bloudek je veliko delal s tem strojem, ena od predelav je tako bila Schwalbe - lastovka, s katero so leta 1911 dosegli avstrijski hitrostni rekord: 163 km/h. Lastovka se je od osnovnega letala razlikovala predvsem po polkrožni obliki kril, ki pa je poleg prednosti predstavljala slabost zaradi zahtevnejše gradnje, zato jih je bilo narejenih le malo. Sledilo je delo v ekipi, ki se je lotila izdelave prvega letala s popolnoma zastekljeno kabino - Luftlimousine. S tem tri sedežnim naprednim letalom so v Angliji avgusta 1912 postavili dva svetovna hitrostna rekorda, najprej z dvema članoma posadke,

112 km/h, in nato še s tremi, 106 km/h.

Pri delu na letalih Taube, oziroma Rumpler Taube, kot je tudi znano, se je prvič srečal tudi z zahtevami po letalu kot orožju. Treba je namreč vedeti, da je prav Taube letalo, ki je leta 1911 kot prvo v zgodovini, ki je bilo uporabljeno kot bojni stroj. Ko je svet obšla novica, da so Italijani bombardirali sovražnika z letalom, je takoj seveda hotela imeti tovrstno orožje tudi Avstro-Ogrska. Bloudek je dobil nalogo, da letalo trdnostno ojača in omogoči njegovo uporabo za vojsko. Že leta 1913 so prodali prvih osem letal, nato pa so začela prihajati naročila od vsepovsod,Nemčije, Francije, Italije, Rusije in celo Kitajske. Bloudek, ki je bil poln zamisli pa se je razšel z Etrichom v idejah o nadaljnjem snovanju novih letal. V svetu so ravno tedaj začeli svojo prevlado dvokrilniki in Bloudek se je nameraval prav tako preusmeriti medtem, ko je njegov delodajalec vztrajal pri enokrilnikih. V tem naj bi bil razlog, da je julija 1913 Bloudek ostal brez službe, kajti dobil je odpoved. Vendar pa kljub svojim 23-im letom ni imel težav pri iskanju nove saj je bil že dovolj priznan v svoji branži.

Bloudek za časa Kraljevine SHS in Jugoslavije ni nikoli preveč oglaševal svojega dela na področju letalstva v času prve svetovne vojne, seveda iz razumljivih razlogov. Šele kasnejše raziskave ob pripravi Bloudkove spominske razstave v okviru Tehniškega muzeja Slovenije so osvetlile njegovo področje dela v vojni industriji tistega časa. Najprej je delal na področju priprave proizvodnje v tovarni letalskih sklopov Thoene & Fiala na Dunaju. Od konca leta 1914 do maja 1916 je bil konstruktor v Dunajskem Novem mestu, in sicer v podjetju Oesterreichische Flugzeugfabrik A.G., skrajšano Oeffag. Tovarna, ki ni imela kakega posebno uspešnega lastnega letala, je kmalu prešla na izdelavo po nemški licenci; delali so takrat razvpite lovce Albatros D II in D III.

Ker mu izdelava tujih letal ni predstavljala nekega posebnega izziva, se je maja 1916 preselil v Albertfavlo pri Budimpešti, kjer je kot višji inženir prevzel vodstvo razvojnega oddelka letalske tovarne Ufag - Ungarische Flugzeugfabrik AG. Leta 1917 je bila končana konstrukcija njegove ekipe, bojno

dvosedežno letalo Ufag C I, ki je bilo dokaj uspešno, saj so ga izdelovali do konca vojne (skupaj jih je bilo 126). Stanko Bloudek je nato začel delati na enosedežnem lovskem letalu (Ufag D I), ki bi naj bil enakovreden drugim takratnim lovcem, toda zaradi razpada Avstro-Ogrske monarhije so se vsa dela na njem ustavila še pred izdelavo prototipa. Za časa njegovega dela v Ufagu se je ukvarjal še z dvema omembe vrednima projektoma. Prvi je zrakoplov kakršnega danes poznamo z imenom helikopter, ki ga omenjajo z imenom Bloudek - Balaban helikopter. Trojka Bloudek, Karl Balaban in Bela Oravecz je sodelovala leta 1917 pri eksperimentu na helikopterju s šestimi kraki na navpično postavljeni osi. Toda zaradi težav s stabilnostjo so se morali projektu odpovedati. Poleg tega pa je omenjeno njegovo delo na mehanizmu delovanja sinhronega streljanja z dvema mitraljezoma v sodelovanju s pogonskim vijakom motorja.

Po prvi svetovni vojni je sodeloval z ljubljanskim aeroklubom pri izgradnji enokrilnega motornega letala Sraka. Letalo so gradili v letih 1924-1925. V letih 1928-1929 pa so v istem aeroklubu izdelali še enokrilno dvosedežno letalo z imenom Lojze. Obe letali so naredili po Bloudkovih načrtih, vendar pa je s tem tudi zaključil gradnjo letal, predvideva se, da zaradi nesreče Lojzeta leta 1934, ko se je med nastopom v Zagrebu z njim ubil pilot Janko Colnar. Takrat se je končalo njegovo obdobje ukvarjanja z letali, čeprav nikoli ne povsem.

== Avtomobilizem ==

Še dosti preden se je zaključila njegova letalska pot se je že začela nova. Po koncu prve svetovne vojne je postal avtomobil preizkušeno prevozno sredstvo in samo vprašanje časa je še bilo, kdaj bo prišlo do njegove množičnosti. Tako je tudi na slovenskem prišlo do prvih delavnic, ki so predelovale vozila in z eno tako, delavnico Jožefa Peterce je sodeloval Bloudek. Poleg tega pa se leta 1926 in 1927 udeležil dirk na prelazu Ljubelj, kjer je poleg nastopanja bil tudi član tehnične komisije dirke. Leta 1933 pa se je pridružil delniški družbi Automontaža, kjer je postal solastnik in glavni konstruktor. Podjetje se je ukvarjalo predvsem s predelavo avtobusov, pa tudi z izdelavo karoserij za

tovornjake in druga lažja vozila. Motorje in podvozja so kupovali pri Mercedesu, Henschlu, Deutzu in drugih ter jih nadgrajevali z lastnimi karoserijami, prirejenimi za domače potrebe in želje posameznih naročnikov.

Poleg tega je Bloudek začel razvijati prvi domači osebni avtomobil, imenovan po Trigavu. Avtomobil je nastal na osnovi nemškega DKW-ja z njihovim dvotaktnim motorjem, ki je poganjal prednji kolesi. Vozilo je stalo 40.000 takratnih dinarjev in bi se lahko še pocenilo ob serijski proizvodnji in še večji udeležbi domačih sestavnih delov, ki je že pri prototipu dosegala polovico vseh. V Automontaži so si zelo prizadevali za osvojitev izdelave v celoti in so začeli celo z razvojem lastnega motorja, ki bi bil sicer podoben DKW-jevemu, le da naj bi bil zračno hlajen. Zakaj in kako so načrti zamrli ni znano, kljub temu da so izdelali več Triglavov, prvi prototip so predstavili v aprilu 1934, med drugimi je obstajal tudi dostavni model za podjetnike.

Automontaža se je leta 1936 preimenovala v Avtomontaža, Bloudek pa je deloval v njej do začetka druge svetovne vojne. Podjetje samo je pozneje delovalo (vmes nekaj časa pod imenom Karoserija) še vse do leta 2002, ko je z njenim stečajem ugasnilo najstarejše avtomobilsko podjetje v Sloveniji.

== Razvoj športa na Slovenskem ==

Stanko Bloudek je pionir mnogih športov na Slovenskem. Leta 1909 je sodeloval pri ustanovitvi nogometnega kluba Hermes. V Ljubljano je prinesel tudi prvo pravo nogometno žogo in nogometne čevlje. Po prvi svetovni vojni je deloval pri športnem društvu Ilirija in bil med vodilnimi možmi tega kluba vse do leta 1941. Med letoma 1925 in 1929 je bil med prvimi slovenskimi nogometaši, slovenski prvak v metu diska ter večkratni prvak Jugoslavije v umetnostnem drsanju. Kot organizator in mecen, trener, svetovalec je zaslužen za začetke in razvoj atletike, tenisa, plavanja, kotalkanja, drsanja, hokeja na ledu, sabljanja, nogometa, smučanja in seveda smučarskih skokov.

== Izgradnja športnih objektov ==

Stanko Bloudek je bil tudi projektant in graditelj več športnih objektov. V Ljubljani je za potrebe Ilirije zgradil, tudi s svojimi sredstvi prvo moderno nogometno igrišče, igrišča za tenis, olimpijski bazen, drsališče in atletsko

stezo. Bloudek je skonstruiral tudi do tedaj eno največjih smučarskih skakalnic na svetu v Planici, ki danes nosi ime Bloudkova velikanka. Od leta 1932 je naredil načrte za okoli 100 smučarskih skakalnic. Z načrtovanjem letalnic je teoretično dokazoval možnost smučarskih poletov in s tem bistveno prispeval k uveljavitvi poletov kot nove športne tekmovalne discipline. Po njegovih načrtih so se zgledovali tudi graditelji skakalnic v Nemčiji in Avstriji. Med drugim je konstruiral tudi Planiško skakalnico, žičnico na Krvavec, žičnico na Mariborsko Pohorje in sodeloval pri izdelavi načrtov za žičnico na Črnem vrhu pri Jesenicah.

Med drugo svetovno vojno je sodeloval z OF, za partizansko vojsko je konstruiral orožja in orodja. Zaradi tega je bil preganjan in zaprt. Po osvoboditvi je spet deloval v vodstvih športnih organizacij. Bil je tudi predsednik fizikalnega odbora Slovenije in vodja projektivnega biroja športne zveze Slovenije. Leta 1948 je postal član Mednarodnega olimpijskega komiteja. Umrl je v Ljubljani 26. novembra 1959. Po njem se imenuje največja, vsakoletna nagrada na področju športa v Sloveniji: Bloudkova priznanja.

== Bloudkova priznanja in imenovanja ==

1919 - izvoljeni član jugoslovanskega olimpijskega komiteja

1934 - SK Ilirija poimenuje po njemu skakalnico v Planici

1936 - Red jugoslovanske krone - IV. stopnje

1937 - Red sv. Save, IV. stopnje

1947 - 1951: predsednik jugoslovanskega olimpijskega komiteja

1948 - član MOK, Mednarodni olimpijski komite

1950 - Red dela, III. stopnje

1953 - Spominska značka, priznanje za delo v Planici

1957 - Cortina d'Ampezzo, diploma OK - ZOI (Organizacijski komite zimskih olimpijskih iger)

1965 - ustanovitev slovenske športne nagrade Bloudkove nagrade, najvišjega priznanja za dosežke na področju športa

1969 - odkritje kipa v spominskem parku v Tivoliju, delo kiparja Stojan Batiča, leta 2008 ukraden

## A.2 Egipčanski hieroglifi

(Vir: Wikipedia [17])

Etimologija ==

Beseda hieroglif izhaja iz grške besede hieroglúfos; hiero- - svet + glúfein - rezbariti. Tradicionalno egipčansko ime za hieroglife mdw.w nṯr se v latinici zapiše kot medu nečer, kar pomeni 'božje besede'. Tako so Grki, ki so obiskali Egipt, imenovali egipčanske ideografe okoli leta 300 pr. n. št.

== Izvor ==

Egipčani so verjeli, da je pisavo izumil njihov bog modrosti Tot in zato so jih imenovali zḫ3 n.j mdw.w nṯr žapis božjih besed". Ker so besede prihajale od bogov, so imele čarobno moč. Iz napisov v grobnicah je razbrati prepričanje Egipčanov, da se z ohranjanjem imen ohranja v živem spominu tudi osebo samo. Če se napis grobnice po nesreči zbriše, izgine tudi oseba. Egipčanski faraoni so imeli po pet imen, od katerih sta bili najpomembnejši kronsko in rojstno ime.

== Zgodovina in razvoj egipčanskih hieroglifov ==

Veliko let je bila najstarejši znan hieroglifski napis paleta faraona Narmerja iz okoli leta 3000 pr. n. št. (tudi 3200 pr. n. št.), ki so jo našli leta 1898 med izkopavanji kraljevskih prostorov v Hierakonpolisu (Neken, danes Kom El-Ahmar). Leta 1987 je nemška arheološka skupina pri izkopavanjih v Abidosu, (danes Um el-Kva'ab), odkrila grobnico U-j, ki je pripadala predinastičnemu vladarju. V njej je bilo več sto kostnih tablic s popolnoma razvitimi hieroglifi. Starost grobnice so ocenili na okoli 3150 pr. n. št.

Hieroglife lahko delimo na tri kategorije znakov:

znake za posamezne glasove (fonograme) kot pri abecedi, a zapisovali so samo soglasnike (razen posameznih samoglasnikov kot npr. a, i ter u, s tem da sta poslednja zapisana kot j in v oz. w). Fonogrami so imeli vrednost enega, dveh ali treh soglasnikov. Po konvenciji egiptologov se na mestih kjer manjka vmesni samoglasnik (tak zapis se uporablja tudi danes npr. v arabščini; bralec mora sam dodati prave samoglasnike) poenoteno izgovarja vmesni "e". (nfr = nefer) ideografe, ki so samostojno predstavljali posamezne besede

(npr. slikovni znak sistruma pomeni sistrum in se fonetično izgovarja sššt, kar pomeni sistrum) determinative, ki so naznačevali razred (oz. kategorijo) posamezne besede brez njenega natančnega pomena (npr. osebe in poklice, bogove, dele telesa, živali, rastline, pohištvo itd). Slednji so bili namenjeni določanju točnega pomena besede, saj je kar nekaj enakih besed z različnimi pomeni zvenelo enako. Pisava se je razvijala glede na način kako, kje in s kakšnim namenom so jo uporabljali. V različnih časovnih obdobjih je bilo sočasno v uporabi več različnih načinov pisave.

Z uporabo papirusa in črnila se je razvila kurzivna pisava hieroglifov. Skozi posamezna časovna obdobja lahko dobro sledimo razvoju te poenostavljene črkovne oblike pisave, ki je na začetku še zelo slična hieroglifski predlogi. Razvoj je šel naprej, iz kurzivne (ležeče tiskanih pismenk) hieratske (svečeniške) pisave je nastala še enostavnejša demotska (ljudska, razširjena) pisava. Obe obliki sta se uporabljali za zapis na papirusu. V hieratski pisavi je ohranjenih veliko osebnih virov, kot so zasebna pisma, zgodbe in poslovne pogodbe, v demotski pa prevladuje zapis pravnih listin. Na znamenitem kamnu iz Rosette (Rašid) iz leta 196 pr. n. št. je identično besedilo zapisano v grške pisave ter tudi še v hieroglifski in demotski pisavi oz. v srednjeegipčanskem in demotskem jeziku.

Hieroglife so uporabljali tudi pod perzijsko nadvlado (s presledki v 6. in 5. stoletju pr. n. št.), po zavzetju Egipta Aleksandra III. Velikega ter med makedonskim in rimskim obdobjem. V kasnejšem času se je začela uporabljati cela vrsta dodatnih hieroglifov (predvsem naraste število determinativov), poleg tega se je začelo uporabljati dodaten zašifriran (skrivnosten) pomen hieroglifov, tako da je sistem branja postal vse bolj zapleten in v njem lahko vidimo neke vrste odziv na spremenjene politične razmere. Hieroglifi so pričeli prevzemati vlogo ločevanja 'pravih Egipčanov' od zunanjih osvajalcev (in njihovih krajevnih lakajev). Tudi ohranjene omembe grških in rimskih avtorjev o hieroglifih podpirajo to tezo.

Do 4. stoletja je lahko bralo hieroglife le nekaj Egipčanov, vse bolj pa je naraščal »mit« o hieroglifih. Leta 391 je rimski cesar Teodozij I. Veliki zaprl

vse nekrščanske templje in raba hieroglifov na spomenikih je zamrla. Zadnji znani zapis je iz templja prav na jugu malo po letu 391.

Iz 4. stoletja je ohranjen Horapollonov spis Hieroglyphica, »razlaga« skoraj 200 znakov. Delo je bilo sicer ugledno, vendar v veliki meri napačno, in je v precejšni meri in za kar nekaj časa oviralo dešifriranje egipčanske pisave. Zgodnejši učenjaki so poudarjali njegov grški izvor, sodobnejši pogled pa nakazuje ostanke pristnega znanja in ga zavrača kot »obupen« poskus egipčanskih razumnikov za rešitev nepovrnljive preteklosti. Hieroglyphica je imela velik vpliv na renesančni simbolizem, še posebej na knjigo podob Emblemata italijanskega pravnika Andree Alciata v zgodnjem 16. stoletju in na delo Hypnerotomachia Poliphili (Polifilov boj za ljubezen in sanje) italijanskega dominikanskega duhovnika in meniha Francesca Colonne iz leta 1499.

Veliko sodobnih učenjakov je skozi stoletja poskušalo dešifrirati hieroglife. Med njimi je bil tudi Athanasius Kircher, ki je med letoma 1650 in 1654 v Rimu izdal štiri zvezke prevodov hieroglifov. Od teh ni bil pravilen niti eden in niti približno ni zadel pomena. Vseeno pa je njegovo delo pomembno, ker je spoznal, kakšno vrednost pomeni učenje koptščine, najkasnejše oblike egipčanskega jezika, ki so jo drugi poskušali utajiti. Poskusi prvih učenjakov niso uspeli. Francoski orientalist Joseph de Guignes je sto let kasneje celo izjavil, da so bili Kitajci egipčanski naseljenci. Poleg Kircherja je egipčanske hieroglife raziskoval tudi francoski učenjak Nicolas Freret. Angleški škof in kritik William Warburton je ugotovil, da egipčansko pisavo sestavljajo tudi znaki v smislu abecede. Pri razvozlanju hieroglifov sta največ prispevala angleški fizik Thomas Young in francoski jezikoslovec Jean-François Champollion, ki je hieroglife dokončno razvozlal med letoma 1822 in 1824. Odkritje kamna iz Rosette leta 1799 članov Napoleonovega vojaškega pohoda na Egipt je Champollionu pripomoglo k dokončnemu razvozlanju sistema pisave in tako predstavlja odločilen korak ter velik uspeh mlade vede, egiptologije.

== Pisava ==

Glavni članek: Egipčanski jezik.

To je zapleten sistem, pisava, ki ima hkrati prenesen, simboličen in glasovni pomen, v istem besedilu, v istem reklu. Skoraj bi rekel, da v isti besedi. Champollion v pismu Dacierju, 27. september 1822.

Hieroglifska pisava ima 24 glavnih enoznačnih znakov, ki predstavljajo samostojen glas, podobno kot slovenske črke, kakor tudi veliko dvoznačnih znakov, ki predstavljajo dva glasova. Obstajajo tudi troznačni glasovi, čeprav v pisavi niso tako pogosti. Vedeti moramo, da večine samoglasnikov v hieroglifski pisavi niso zapisovali, tako da pri izgovarjavi med soglasniki dodajamo »e«. Na primer: nfr -¿ nefer - lep, dober.

Drug primer kako se lahko hieroglife razume je pomen egipčanske besede pr (po navadi izgovorjeno kot per). Prvi pomen je bil 'hiša', hieroglifski zapis je bil enostaven:

Tukaj je hieroglif 'hiša' deloval kot ideogram: predstavljal je besedo z enim samim znakom. Z navpično potezo pod hieroglifom so v splošnem označili, da ima znak pomen ideograma.

Beseda pr je lahko pomenila tudi 'iti ven, oddti'. Kadar so zapisali to besedo, je hieroglif 'hiša' pomenil glasovni znak: Tukaj je hieroglif 'hiša' pomenil soglasnika pr. Hieroglif 'usta' spodaj je bil glasovno dopolnilo: bere se kot r, in ojača glasovno izgovorjavo pr. Tretji hieroglif je bil determinativ, ideogram, ki je podal bralcu širok pomen tistega kar je bilo zapisano. Tukaj je nakazoval glagol gibanja. Ker je bilo pri takšni pisavi možno, da je imelo zaporedje znakov več pomenov, lahko vidimo, kako težko je bilo razvozlati hieroglife.

## A.3 Vanilija

(Vir: Wikipedia [41])

Glavna sorta za pridelavo vanilina je Vanilla planifolia. Četudi prvotno prihaja iz Mehike, danes uspeva v vseh tropskih predelih. Madagaskar je največji svetovni pridelovalec vanilje. Tudi vrsti Vanilla pompona in Vanilla tahitiensis (raste na Tahitiju) vsebujeta vanilin, a precej manj kot Vanilla

planifolia. Vanilja raste kot vinska trta, vzpenja se po drevesu, palici ali drugi podpori. Lahko raste v gozdu (na drevesih) ali na plantažah (na drevesih ali palicah).Brez nadzora raste v višino in skromno cveti, zato jo pridelovalci upogibajo navzdol in s tem pospešujejo cvetenje.

Značilna dišava je v samem plodu, ki je rezultat oprašitve cveta. En cvet proizvede en plod. Cvetovi Vanilla planifolia so hermafroditi, dvospolniki; imajo tako moške (prašnik) kot ženske (pestič) rastlinske organe, ki jih ločuje membrana in s tem preprečuje oprašitev. Belgijski botanik, Charles François Antoine Morren, je odkril, da so lahko cvetovi naravno oprašeni le preko specifične čebele Melipone, ki jo najdemo v Mehiki. Gojitelji so skušali to vrsto čebele prenesti na ostale lokacije, kjer uspeva vanilja, a brez uspeha. Edini način proizvodnje plodu je tako umetna oprašitev.

Preprosta in učinkovita metoda umetne oprašitve je bila vpeljana (predstavljena) leta 1841. Metodo, ki se uporablja še danes, je predstavil 12-letni suženj Edmond Albius iz Réuniona (otok v indijskem oceanu, vzhodno od Madagaskarja). Z bambusovo trsko se membrano zaviha nazaj tako da se prašnika in pestič ločita, nato pritisnejo prašnik na pestič. Tako je cvet samooprašen in razvije plod. Cvet vanilje cveti približno en dan, včasih tudi manj, tako morajo gojitelji vsak dan nadzorovati plantažo v pričakovanju cvetu, kar pa je mučna in intenzivna naloga. Če plod (strok) ostane na rastlini, se odpre in sprosti značilen voln po vanilji. Plod vsebuje majceno seme. V jedi pripravljeni z naravno vaniljo, je to seme prepoznavno kot črn 'madež'.

Kot ostala semena orhidej, tudi vanilja ne bo kalila brez prisotnosti določenih mikoriznih gliv. Namesto tega gojitelji reproducirajo rastlino s potaknjenci– odstranijo del plezalke s šestimi ali več listnatimi vozli. Dva najnižja lista odstranijo in pecelj z zračno korenino zakopljejo v zemljo blizu opore. Ostale višje korenine so pripete, ovite ob podpori in pogosto rastejo navzdol, v zemljo. Ob dobrih pogojih je rast hitra.

== Zgodovina ==

Prvi so gojili vaniljo Totonaci, ljudstvo, ki je prebivalo v dolini Mazantla,

v mehiškem obalnem zalivu, današnji Veracruz. Legenda pravi, da je bila tropska orhideja rojena, ko je princesa Xanat pobegnila v gozd s svojim ljubimcem, saj ji je oče prepovedal poroko s smrtnikom. Zaljubljenca so ujeli in ju obglavili. Kjer je njuna kri prišla v stik z zemljo, je zrasla tropska orhideja.

V 15. stoletju so Azteki iz centralnega višavja Mehike zavzeli Totonace in se preko njih spoznali z vaniljo. Strok so poimenovali /tlilxochitl/ ali črna roža, saj se po obiranju hitro zguba in postane črn. Totonaci so Aztekom plačevali redni davek v obliki vaniljevih strokov. Vanilja je bila povsem nepoznana staremu svetu pred Kolumbom. Španski osvajalci, ki so v zgodnjih letih 16. stoletja prispeli v Mehiški zaliv, so skupaj s portugalskimi pomorščaki prenesli vaniljo v Afriko in Azijo. Poimenovali so jo vainilla. Do sredine 19. stoletja je bila Mehika glavni proizvajalec vanilje. Leta 1819 so Francozi po morju prepeljali stroke vanilje na Reunion in otoke Mavricija v upanju, da bodo lahko tam gojili vaniljo. Edmond Albius, 12 – letni suženj z Reunionskih otokov, je odkril hitro ročno metodo oprašitve in rastlina je začela uspevati. Kmalu je bila tropska orhideja z navodili oprašitve poslana na Komorske otoke in na Madagaskar. Do leta 1898 so na Madagaskarju, v Reunion in na Komorskih otokih pridelali 200 ton vaniljinih strokov, kar je približno 80% svetovnega pridelka.

Cena vanilje je drastično narasla v poznih 1970-ih, zaradi tajfuna in držala ceno do zgodnjih 1980-ih, navkljub konkurenci indonezijske vanilje. Monopolisti, ki so nadzirali ceno in distribucijo vanilje od leta 1930, so se v sredini 80-ih razšli. Cene so v naslednjih nekaj letih padle za 70%, nekako na 20$/kg. To pa se je spremenilo po tajfunu Huddah, ki je udaril v letu 2000. Tajfun, politična nestabilnost in slabo vreme so v privedli do presenetljive cene 500$/kg, v letu 2004. Dober pridelek in padec povpraševanja pa sta pripeljala do padca tržne cene na 40$/kg v sredini leta 2005.

K temu je veliko pripomogla tudi pridelava umetne arome vanilje. Madagaskar (predvsem rodovitna regija Sava) ocenjuje svoj pridelek na polovico pridelave vanilje na svetovnem trgu. Mehika, nekoč vodilna pridelovalka na-

ravne vanilje, letno 500 ton, pa je v letu 2006 pridelala le še 10 % vanilje na svetovnem trgu. Ocenjenih 95% izdelkov z aromo vanilje dejansko vsebuje umetni vanillin, pridelan iz lignina.

== Kemija ==

Četudi ima ekstrakt vanilje številne sestavine, je za značilnosti in vonj vanilje primarno odgovoren vanilin (4-hidroksi-3-metoksi-benzaldehid). Manjši delež sestave vanilje je eterično olje piperonal (heliotropin). Piperonal in ostale substance dajejo vonj naravne vanilje. Vaniljin izvleček obstaja v dveh oblikah. Ekstrakt pravega semenskega stroka je izredno zapletena mešanica stotin različnih sestavin. Sintetični izvleček, ki v osnovi vsebuje raztopino sintetičnega vanillina v etanolu, prihaja iz fenola in je zelo čist.

== Faze proizvodnje ==

Pridelek Stroke se nabira, ko so zeleni in še niso zreli. V tej fazi so brez vonja.

Uničenje

Vegetativno tkivo vanilje se uniči, da se prepreči nadaljnja rast. Metode uničenja so različne, najpogosteje s pomočjo sonca, s sterilizacijo z vročim zrakom, uničenje z vročo vodo, s praskanjem ali z zamrznitvijo.

Izkoriščanje

Stroke se nato 7 do 10 dni izpostavi vročini (45-65 stopinj celzija) in vlago - običajno se jih položi v pokrite škatle takoj po uničenju vegetativnega tkiva. To omogoči encimom, da sestavine v stroku pretvorijo v vanilin in druge snovi, ki dajejo vanilji značilen vonj.

Sušenje

Da preprečijo gnitje in zadržijo arome v stroku, jih posušijo. Pogosto se stroki čez dan sušijo zunaj na soncu, popoldne pa jih vrnejo v škatle. Ko vlaga predstavlja le še 25-30
% teže stroka (v nasprotju s 60-70% ob začetku sušenja), le-ta doseže optimalno strukturo za rabo v kulinariki.

Razvrščanje

Posušene stroke vanilje razvrstijo po kvaliteti, glede na aromo.

== Raba v prehrani ==

Obstajajo tri glavne tržne priprave naravne vanilje:

celi stroki,

prašek (zemeljski strok, čist ali zmešan s sladkorjem, škrobom ali ostalimi sestavinami),

ekstrakt (alkoholna raztopina).

Okus vanilje lahko v prehrani dosežemo z dodajanjem ekstrakta vanilje ali s kuhanjem strokov vanilje . Intenzivnejša aroma se doseže, če se stroke razdeli na dva dela, razstavljanje večje površine strokov v tekočino. V tem primeru se semena zmešajo v pripravek. Naravna vanilja da pripravki rjavo ali rumenkasto barvo, odvisno od koncentracije.

Visoko kvalitetna vanilja ima močan aromatičen okus. Vendar pa prehrana z vaniljo nizke kvalitete ali z umetno vaniljo bolj običajna, saj je prava vanilja veliko dražja.

Najpogostejša raba vanilje je v namen odišavljenja sladoleda. Najpogostejši okus sladoleda je vanilja, zato večina ljudi meni, da je okus vanilje osnoven okus sladoleda. O analogiji se termin vanilla zato včasih uporablja kot sinonim za osnovno, enostavno ali navadno. Kozmetična industrija pa uporablja vaniljo za izdelavo parfumov.

Kulinarična industrija uporablja metil in etil vanillin. Etil vanilin je dražji, vendar ima močnejši vonj. V Cook's Ilustrated (kuharska revija) so prikazali različne okuse testiranja vanilje in vanillina v pečenih in drugih dobrotah. Na presenečenje urednikov revij okuševalci niso mogli ločiti med okusom vanillina in anilje. Kakorkoli, v primeru vaniljnega sladoleda pa naravna vanilja zmaga.

== Medicinski učinki ==

V stari medicinski literaturi je vanilja opisana kot afrodiziak in zdravilo za vročice. Slednji rabi nista znanstveno dokazani, vendar pa je bilo prikazano, da vanilja povzroči rast stopnje kateholaminov (vključujoč epinephrine, bolj znan kot adrenalin) in kot takšen je lahko smatran tudi kot blag povzročitelj odvisnosti.

V in-vitro poskusu je vanilja lahko ustavila občutljivost bakterij na kvorum. To je z medicinskega vidika zanimivo, saj v številnih bakterijah občutljivih na kvorum deluje kot izključitelj za ustavitev virulence. Mikrobi postanejo virulentni le, ko jih je številčno zadosti, da se lahko uprejo imunskemu sistemu gostitelja. Eterična olja vanillin-a in vanilje so včasih uporabljeni v aromaterapiji.

== Vrste vanilje ==

Burbonska vanilja (okorela vanilja) ali bourbon – Madagaskar vanilja, ki se pridela iz vanilla planfolia in je pripeljana iz Amerike. Termin se uporablja za vaniljo z otočij indijskega oceana, kot na primer Madagaskar, Komorski otoki , Reunion, nekoč île Bourbon, torej bourbonski otok, otok Bourbon. Mehiška vanilja, pridelana iz naravne vanilje planifolia, je pridelana v veliko manjših količinah in se jo trži kot originalno vaniljo, saj je pridelana na območju iz katerega izvira. Vanilja, ki jo na tržnicah po Mehiki prodajajo turistom včasih ni pravi ekstrakt vanilje, ampak je zmešan z ekstraktom tonka fižola, ki vsebuje kumarin. Ekstrakt tonka fižola ima vonj in okus kot vanilja, vendar pa je bilo dokazano na laboratorijskih-poskusnih živalih, da kumarin škoduje jetrom.

Tahitijska vanilja poimenujemo vaniljo, ki prihaja iz francoske Polinezije, pridelana pa je iz orhideje sorta Vanilla tahitiensis. Ta vrsta izhaja iz Vanilla plantifolia, ki je bila pripeljana na Tahiti pred mutacijo v različne vrste.

Francoska vanilja ni vrsta vanilje, z izrazom pa poimenujemo preparate, ki imajo močno aromo vanilje in vsebujejo tudi zrna vanilje. Ime izhaja iz francoskega načina priprave sladoleda z okusom jajčne kreme, ki temelji na vaniljinih strokih, smetani in rumenjakih. Francoska vanilja se nanaša na okus vaniljeve jajčne kreme. Sirup francoske vanilje lahko vsebuje jajčno kremo, karamelo (žgani sladkor) ali mehko masleno karamelo, kot dodatek vanilji.

# Dodatek B

# Primeri originalnih povzetkov

## B.1 Stanko Bloudek povzetek

(Vir: Wikipedia [40])

Stanko Bloudek, slovenski letalski konstruktor, športnik, načrtovalec športnih objektov, * 11. februar 1890, Idrija, † 26. november 1959, Ljubljana.

## B.2 Egipčanski hieroglifi povzetek

(Vir: Wikipedia [17])

Hieroglífi so pisava, ki so jo uporabljali v starem Egiptu. Pisava je uporabljala kombinacije samoznakov (logografov) z zlogovnimi (silabičnimi) in abecednimi elementi.

## B.3 Vanilija povzetek

(Vir: Wikipedia [41])

Vanilja oz. vanilija je posušen in fermentiran strok rastline Vanilla planifolia, iz rodu kukavičevk, ki izvira iz Mehike. Vanilja je cenjena zaradi sladkega okusa in prijetnega ter izrazitega vonja, v veliki meri se uporablja

za pripravo sladic in parfumov. Večina svetovne proizvodnje vanilje se nahaja v eni manjših regij otoka Madagaskar v Indijskem oceanu.

# Dodatek C

# Primeri povzetkov našega sistema

## C.1 Stanko Bloudek

Primeri povzetkov generiranih z našim sistemom iz besedila v dodatku A.1 [40].

### C.1.1 Graf utežen z seštevkom pojavitve besede in P-PR vrednost z enakomerno utežjo

Kot otrok od skupno petih imel je kar štiri sestre po dve mlajši in dve starejši. Dunaju je umrla mati. Modele je poimenoval Racek - Galeb. Kot zanimivost velja omeniti da je bil tretji Slovenec ki je delal v tem podjetju. Bloudek je dobil nalogo. Delavnico Jožefa Peterce je sodeloval Bloudek.

### C.1.2 Graf utežen z seštevkom pojavitve besede in P-PR z tf-idf vrednostjo

Je bratranec po materini strani Dolfe Lapajne študiral slikarstvo. Rakom je umrla mati. Dunaju je dobil nalogo. Kasnejše raziskave ob pripravi Bloudkove spominske razstave so osvetlile njegovo področje dela v vojni industriji

tistega časa. Je z nastopom Zagrebu ubil pilot Janko Colnar. Ilirija poimenuje po član komiteja skakalnico v Planici.

### C.1.3 Graf utežen enakomerno in P-PR z okapi bm-25 vrednostjo

Je oče dobil službo v rudniku rjavega premoga. Je bratranec po materini strani Dolfe Lapajne študiral slikarstvo. Dunaju je umrla mati. Modele je poimenoval Racek - Galeb. Je z nastopom Zagrebu ubil pilot Janko Colnar. Ilirija poimenuje po član komiteja skakalnico v Planici.

### C.1.4 Graf utežen z številom dvogramov in P-PR vrednost z pogostostjo besede

Dunaju je umrla mati. Bloudek je delal s tem strojem. Lastovka je od osnovnega letala razlikovala. Bloudek je dobil nalogo. Delavnico Jožefa Peterce je sodeloval Bloudek. Ilirija poimenuje po član komiteja skakalnico v Planici.

## C.2 Vanilija

Primeri povzetkov generiranih z našim sistemom iz besedila v dodatku A.3 [41].

### C.2.1 Graf utežen z seštevkom pojavitve besede in P-PR vrednost z enakomerno utežjo

Višino pridelovalci upogibajo. Plod vsebuje majceno seme. Legenda pravi da je bila rojena. To omogoči encimom da v stroku pretvorijo v vanilin in druge snovi ki dajejo. Večina ljudi meni da je. Kozmetična industrija uporablja vaniljo za izdelavo parfumov. Kumarin škoduje jetrom.

### C.2.2 Graf utežen z seštevkom pojavitve besede in P-PR z tf-idf vrednostjo

Višino pridelovalci upogibajo. Plod vsebuje majceno seme. To omogoči encimom da v stroku pretvorijo v vanilin in druge snovi ki dajejo. Visoko kvalitetna vanilja ima močan aromatičen okus. Večina ljudi meni da je. Kozmetična industrija uporablja vaniljo za izdelavo parfumov. Kumarin škoduje jetrom.

### C.2.3 Graf utežen enakomerno in P-PR z okapi bm-25 vrednostjo

Višino pridelovalci upogibajo. Plod vsebuje majceno seme. Piperonal in ostale substance dajejo vonj naravne vanilje. Visoko kvalitetna vanilja ima močan aromatičen okus. Večina ljudi meni da je. Kozmetična industrija uporablja vaniljo za izdelavo parfumov. Tahitijska vanilja poimenujemo vaniljo ki prihaja iz francoske Polinezije.

### C.2.4 Graf utežen z številom dvogramov in P-PR vrednost z pogostostjo besede

Višino pridelovalci upogibajo. Plod vsebuje majceno seme. Španski osvajalci ki so v zgodnjih letih 16. stoletja prispeli so prenesli vaniljo v Afriko in Azijo. Ima ekstrakt vanilje številne sestavine. Piperonal in ostale substance dajejo vonj naravne vanilje. To omogoči encimom da v stroku pretvorijo v vanilin in druge snovi ki dajejo. Kozmetična industrija uporablja vaniljo za izdelavo parfumov.

# Dodatek D

# Človeški povzetek vanilije

Primer človeškega povzetka iz besedila v dodatku A.3 [41].

Rastlina vanilije, katere cvetovi so hermafroditi, raste v tropskih predelih sveta in se vzpenja podobno kot trta ob opori. Za nastanek plodov je potrebna umetna oploditev cvetov, ki jo je leta 1841 predstavil suženj Edmond Albius z otoka Reunion, saj lahko cvetove naravno oprašijo le čebele Melipone, ki kljub človeškemu trudu, da bi jih preselili tudi drugam, živijo le v Mehiki, od koder tudi izvira vanilija. Za njen vonj in okus je primarno odgovoren vanilin (4-hidroksi-3-metoksi-benzaldehid), manjši delež pa eterično olje piperonal (heliotropin).

Prvi so gojili vaniljo Totonaci, ljudstvo, ki je prebivalo mehiškem obalnem zalivu in so ga v 15.stoletju zavzeli Azteki, ki so kot plačilo davka od ljudstva prejemali stroke vanilije. Vanilija je postala staremu svetu znana šele po prihodu Kolumba na Ameriško celino. Do sredine 19.stoletja je bila Mehika glavna proizvajalka vanilije, nato pa je njeno mesto prevzel Madagaskar. Cena vanilije je v preteklosti nihala zaradi naravnih katastrof in nestabilnosti na Madagaskarju. V stari medicinski literaturi je vanilja opisana kot afrodiziak in zdravilo za vročice. Eterična olja vanillina in vanilje so občasno uporabljena v aromaterapiji.

Faze proizvodnje so pobiranje pridelka, ko je še zelen, uničenje vegetativnega tkiva, ki prepreči nadaljnjo rast, sušenje do primerne vsebnosti vlage

ter razvrščanje glede na aromo. V kulinariki se lahko uporabijo celi stroki, prašek ali ekstrakt. Najpogosteje se vanilijo uporabi za sladoled, zato večina meni, da je to osnoven okus sladoleda in uporablja izraz vanilla kot sinonim za osnovno, navadno. Vrste vanilije so Burbonska vanilija ali burbon, ki se jo pridela na otokih indijskega oceana, Mehiška vanilija oziroma originalna vanilija, saj se jo prideluje v Mehiki ter Vanilla tahitiensis, ki se jo prideluje v francoski Polineziji. Francoska vanilija je samo izraz za preparate z močno aromo vanilije in se nanaša na okus vanilijine jajčne kreme.